ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK

MP

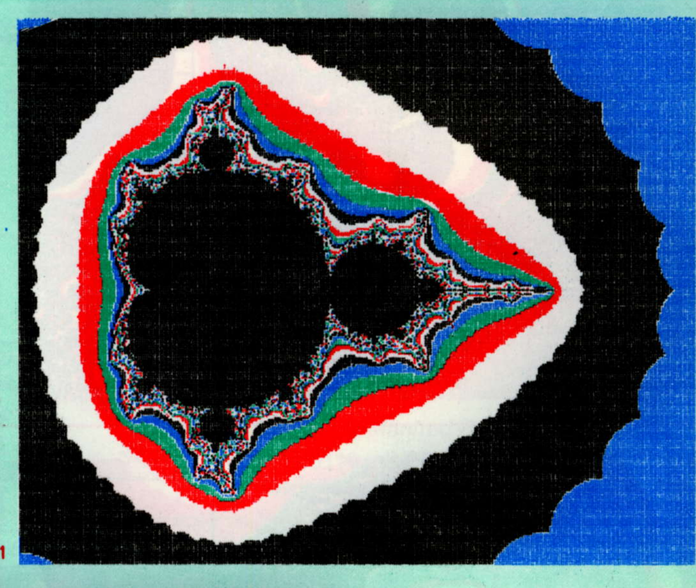
Heft 1 · 1988

Mikroprozessortechnik

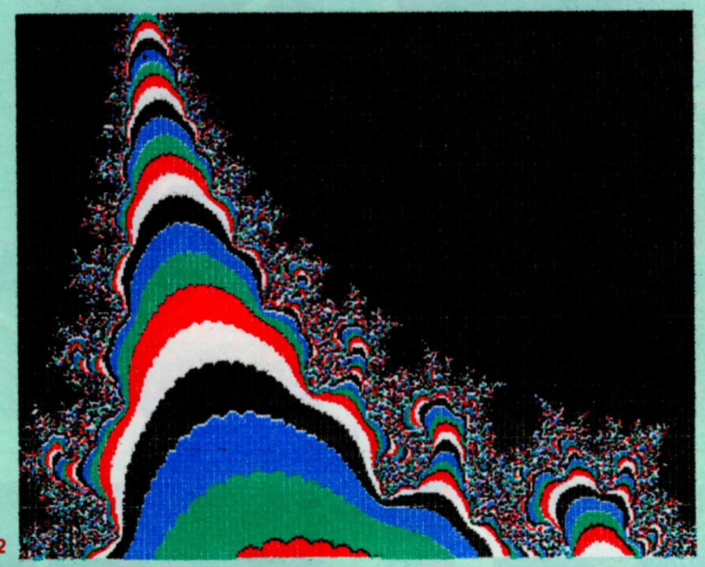
VEB Verlag Technik Berlin

ISSN 0232-2892

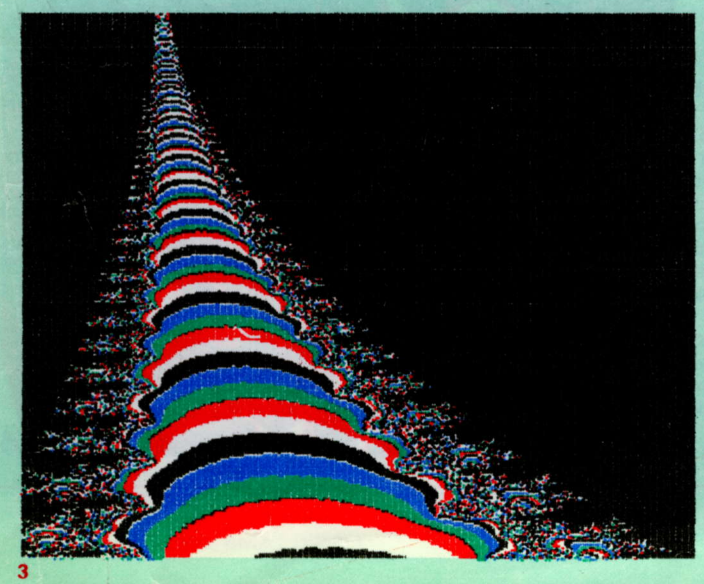
Neue Fraktale



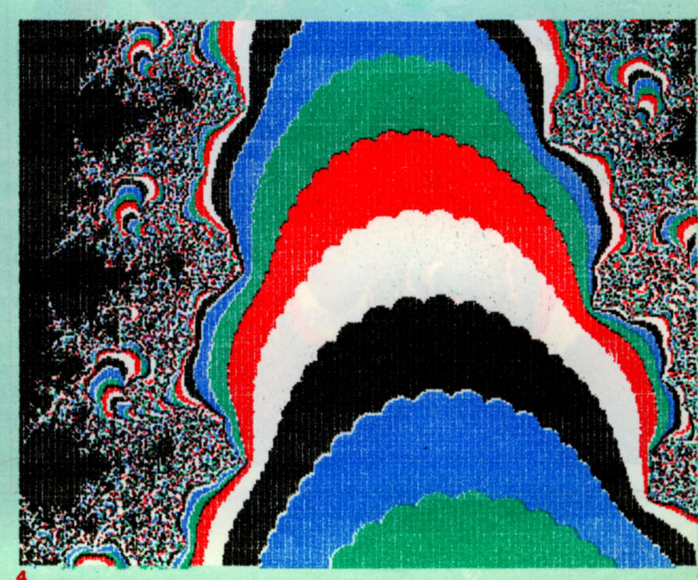
1



2

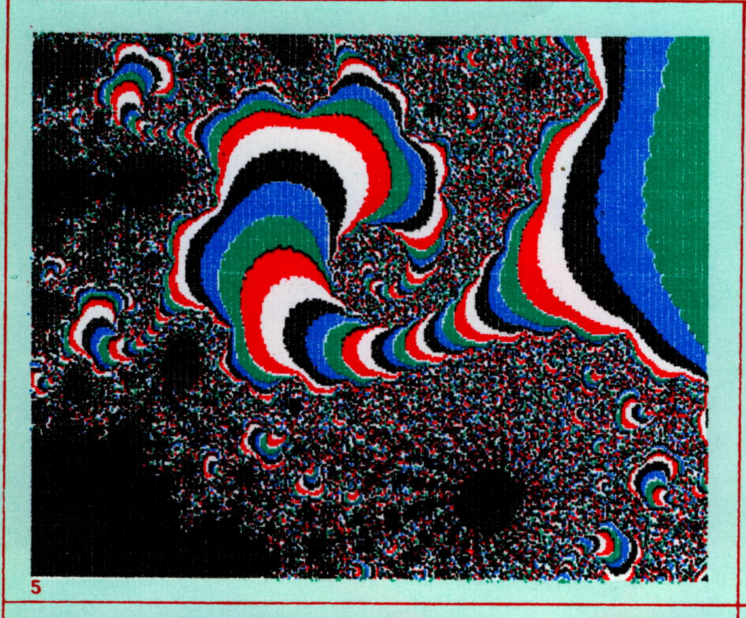


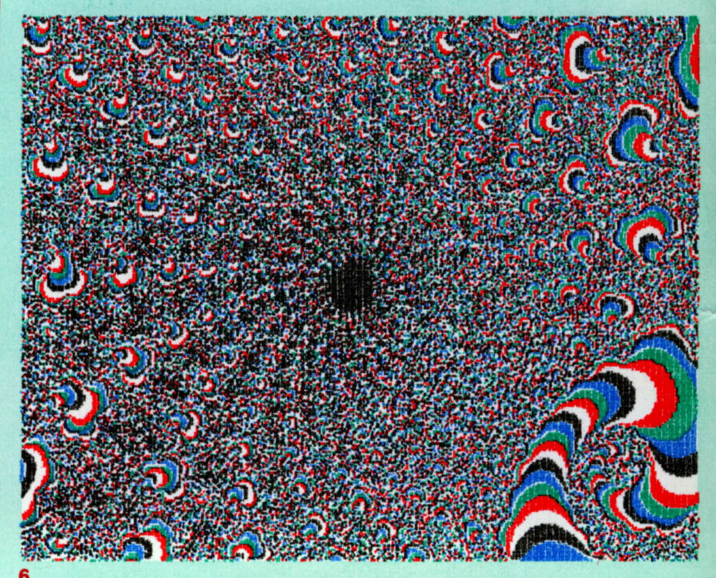
3

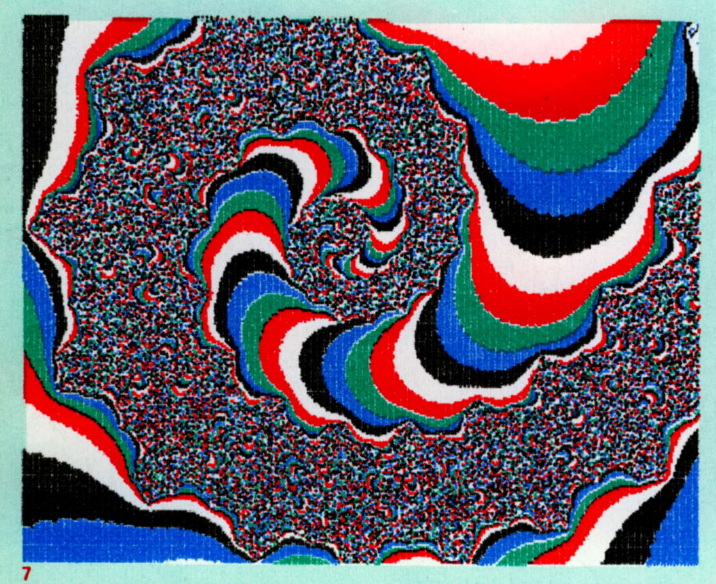


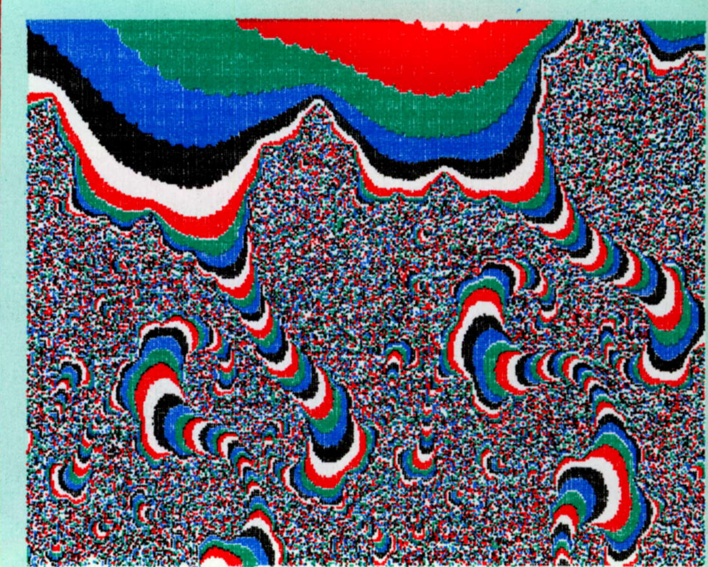
4

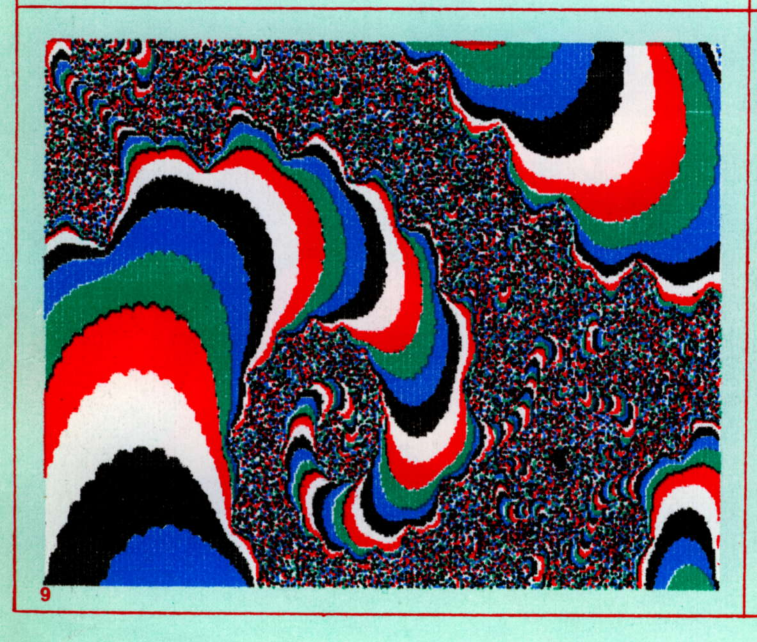
Neue Fraktale

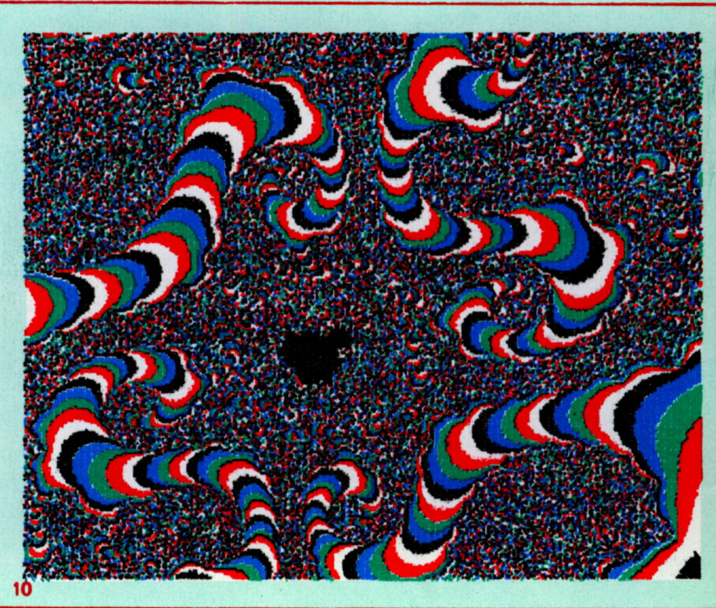

5


6

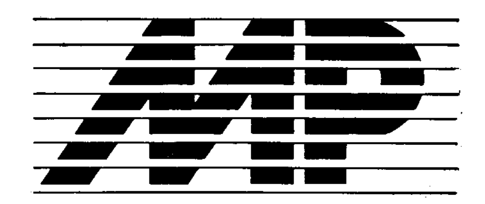

7


8


9


10

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 1 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR–1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Ingo Paszkowsky, Verantwortlicher Redakteur (Tel.: 2870203); Hans Weiß, Redakteur (Tel.: 2870371); Sekretariat Tel.: 2870381

Gestaltung Christina Kaminski (Tel.: 2870288)

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

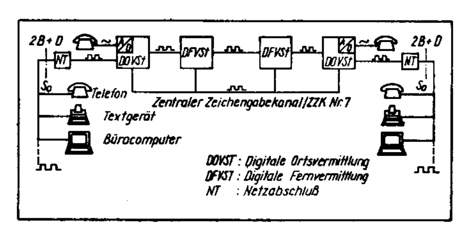
Redaktionsschluß: 12. November 1987

AN (EDV) 49837

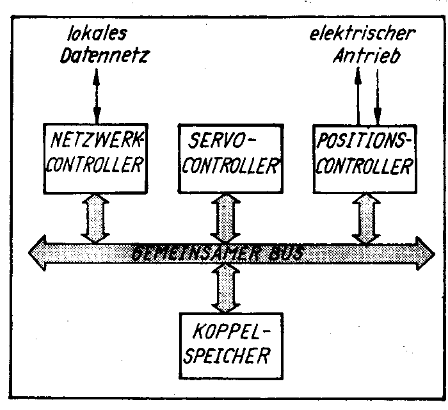
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

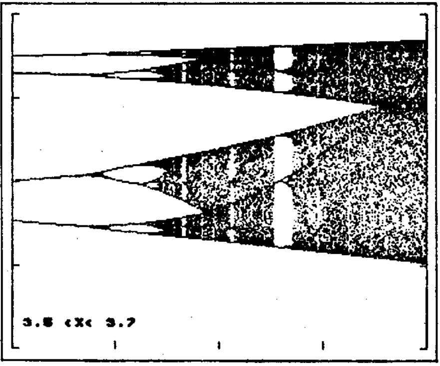
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tísku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Hà Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR–7010, und Leipzig Book Service, Talstraße 29, DDR–7010 Leipzig



Seite 3



Seite 10



Seite 24

Zu unseren Umschlagseiten

Die Farbbilder auf der 3. und 4. Umschlagseite sind die Darstellungen neuer, meist bisher nicht bekannter Fraktale. Sie beruhen auf ähnlichen Iterationen wie beim Apfelmännchen auf der 1. und 2. Umschlagseite. Die zugehörigen Formeln und Parameter werden wir Ihnen in einem der nächsten Hefte mitteilen, darüber hinaus eine Methode, mit der Sie relativ schnell neue Formeln und Gebiete finden.
Wir hoffen, Ihnen mit den beiden Beiträgen dieses Heftes bereits Anregungen zum „ästhetischen Gestalten" mit dem Kleincomputer zu geben. Die hier gezeigten Bilder sollen Sie darauf ein wenig neugierig machen.
Computergrafiken: Prof. Dr. Horst Völz

Inhalt



Dialog

Prof. Dagmar Hülsenberg zum Präsidenten der Kammer der Technik gewählt

Zum neuen Präsidenten der Kammer der Technik hat das Präsidium der Ingenieurorganisation auf seiner Tagung im Oktober vergangenen Jahres in Berlin Prof. Dr.-Ing. Dr. rer. oec. Dagmar Hülsenberg gewählt.



Prof. Dr.-Ing. Dr. rer. oec. Dagmar Hülsenberg wurde am 2. Dezember 1940 in Sonneberg geboren. Nach dem Abitur erlernte sie den Beruf eines Facharbeiters für technische Keramik und studierte ab 1960 an der Bergakademie Freiberg in der Fachrichtung Silikathüttenkunde. Dort promovierte sie auch zum Dr. rer. oec. und zum Dr.-Ing. Nach wissenschaftlicher Arbeit an der Bergakademie Freiberg und mehrjähriger Tätigkeit im Staatsapparat wurde sie 1975 zum Ordentlichen Professor an die Technische Hochschule Ilmenau berufen. Ihre Spezialgebiete in Lehre und Forschung sind Glas- und Keramikwerkstoffe sowie -technologien. Vorwiegend befaßt sie sich in ihrer wissenschaftlichen Tätigkeit mit neuen Technologien der Herstellung und mit der Anwendung von Glas, Keramik und Vitrokeramik in der Mikroelektronik, mit dem Einsatz von Sensoren und der Entwicklung von Robotern für die Glas- und Keramikindustrie sowie der Wissenschaftsorganisation.
Prof. Dr. Dr. Hülsenberg ist Mitglied der SED. Sie wurde 1976 zur Vorsitzenden des Fachverbandes Silikattechnik der Kammer der Technik berufen und 1978 zum Mitglied des Präsidiums der KDT gewählt. Sie ist Mitglied des Forschungsrates der DDR und seit 1986 Ordentliches Mitglied der Sächsischen Akademie der Wissenschaften.
Für ihr verdienstvolles Wirken wurde sie 1986 als „Verdienter Techniker des Volkes" ausgezeichnet. Sie ist Träger der höchsten Auszeichnung der Kammer der Technik, der Ernst-Abbe-Medaille.
Foto: ADN-ZB/Schöps

Harry Pollei zum Sekretär des FV Elektrotechnik berufen

Zum Sekretär des Fachverbandes Elektrotechnik der KDT wurde mit Wirkung vom 1. 7. 1987 Dipl.-Ing., Dipl.-Ges.-Wiss. Harry Pollei (40) berufen.



Der gelernte Kabelfacharbeiter studierte 1965 bis 1970 an der TU Dresden, Sektion Informationstechnik, und wirkte anschließend in einem Chemiebetrieb an der Einsatzvorbereitung von Prozeßrechentechnik mit.
1971 bis 1976 als wissenschaftlicher Mitarbeiter am Zentralinstitut für sozialistische Wirtschaftsführung beim ZK der SED und 1976 bis 1983 als Redakteur der Zeitung „Die Wirtschaft" befaßte sich Genosse Pollei vorrangig mit Leitungsaufgaben und -erfahrungen bei der Anwendung von Rechentechnik und Mikroelektronik in der Volkswirtschaft.
1973 bis 1978 absolvierte er ein Fernstudium an der Parteihochschule „Karl Marx" beim ZK der SED.
Seit 1983 ist Harry Pollei im Präsidium der KDT tätig.

Was darf der Leser von einem Fachartikel erwarten?

Als ständiger Leser von MP und rfe erscheint mir eine öffentliche Meinungsäußerung zu dieser Frage dringend geboten. Ich meine, vor allem hat der Leser ein Recht auf konkrete Informationen. „Konkret" bedeutet in diesem Zusammenhang, daß z. B. ein Beitrag über eine K-1520-Interfacekarte oder einen Einplatinenrechner zumindest den vollständigen Stromlaufplan enthalten muß. Aber auch die Angabe von Leiterbild und passender Software sollte nichts Ungewöhnliches sein.
Was nützen dem Leser mehrseitige Ausführungen zum Thema „Einplatinenrechner mit I 8086"/1/, wenn die Beschreibung des Rechners bei der Aufzählung der verwendeten Peripherieschaltkreise und einem Blockschaltbild endet? Wem nützt die Angabe der Portadressen ohne die genaue Kenntnis der Schaltung? Lediglich diejenigen Leser, die aufgrund ihrer aktuellen dienstlichen Arbeitsaufgaben in der Lage bzw. gezwungen sind, ihren Betrieb einen Nachnutzungsvertrag abschließen und die damit verbundene Gebühr entrichten zu lassen, kommen (endlich) in den Besitz sofort verwertbarer technischer Informationen. In der Summe dürfte diese Gruppe einen Anteil von höchstens 1 % der gesamten Leserschaft ausmachen. Für die übrigen 99 % bleiben Artikel dieser Art unbefriedigend.
Erstes Ziel eines Fachbeitrags in der MP sollte daher nicht die Verfolgung wirtschaftlicher Interessen, sondern die Aus- und Weiterbildung auch und gerade der in der einschlägigen Industrie tätigen Leser sein.
Zum Glück gibt es immer wieder Beispiele, die zeigen, daß es auch anders (nämlich besser) geht, /3/, /4/, /5/ u. a.
Besonders aufschlußreich ist ein Vergleich zwischen /2/ und /5/, da beide Artikel das gleiche Thema, nämlich eine EPROM-Programmierkarte für ein U880-System, behandeln.
Während /2/ wenig konkrete Informationen zur technischen Lösung, wohl aber einen Hinweis zur Nachnutzung enthält, werden in /5/ sowohl der vollständige Stromlaufplan als auch das komplette Treiberprogramm angegeben.
Dem Autor von /5/ kann man für sein offensichtliches Bemühen, den Lesern einen attraktiven Beitrag zu bieten, nur Dank und Anerkennung aussprechen.
Um Mißverständnissen vorzubeugen: Nachnutzungen sind eine sinnvolle und nützliche Einrichtung. Sie dürfen jedoch nicht dazu führen, daß Autoren von Fachartikeln vorsätzlich Informationen zurückhalten und so die Attraktivität unserer Fachzeitschriften leidet. Für Nachnutzungsangebote ohne detaillierte technische Beschreibung gibt es bekanntlich die MP-Rubrik „Börse".

Wilfried Skutsch, Potsdam

Ihrer Meinung können wir nur teilweise zustimmen. Werden Lösungen in dem o. g. Sinn so ausführlich beschrieben, daß sie unmittelbar aus den abgedruckten Informationen nachgenutzt werden können, dann ist dafür recht viel Platz in der Zeitschrift erforderlich. Spezielle Fachbeiträge interessieren in der Regel aber immer nur einen zur Gesamtleserschaft vergleichsweise geringen Leserkreis. Andererseits wird zu je dem Fachartikel eine Kontaktadresse angegeben, unter der Leser, die weitergehende Informationen zur Thematik wünschen, nachfragen können. Die von Ihnen gewählten Beispiele /2/ und /5/ verdeutlichen dies sehr gut. Während in /2/ auf etwas über 2 Druckseiten das Funktionsprinzip und der Leistungsumfang ausführlich erläutert wurden, waren für die Darstellung /5/ 4 Druckseiten erforderlich.
Dennoch wollen wir bei Themen, von denen wir annehmen, daß sie für einen breiten Leserkreis von Interesse sind, die von Ihnen und anderen Lesern vorgeschlagene „Vollständigkeit" zulassen. Dabei legt uns natürlich auch der Umfang von MP eine Begrenzung auf.
Gerne würden wir die Auffassungen weiterer Leser zu dieser Problematik wissen.

Literatur

/1/ Münzer, B.-G.; Stachowiak, T.: 16-Bit-Single-Board-Computer SBC 8086. Mikroprozessortechnik, Berlin 1 (1987) 7, S. 200
/2/ Kabatzke, W.: K-1520-kompatible Programmiereinheit. Mikroprozessortechnik, Berlin 1 (1987) 2, S. 43
/3/ Münzer, B.-G.: Kopplung eines DMA-Schaltkreises an den Mikrorechner K 1520. Nachrichtentech., Elektron., Berlin 32 (1982) 12, S. 499
/4/ Münzer, B.-G.: Kopplung eines programmierbaren Interrupt-Controllers an den Mikrorechner K 1520. Nachrichtentech., Elektron., Berlin 31 (1981) 5, S. 183
/5/ Möckel, F.: EPROM-Programmierer für KC 85/2 und KC 85/3. Radio, Ferns., Elektron., Berlin 36 (1987) 7, S. 419

In eigener Sache

Ab sofort suchen wir für die Stelle eines Redakteurs unserer Zeitschrift MP eine(n) geeignete(n) Mitarbeiter(in) mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik (Hard- und Software).
Zu den Aufgaben gehören u. a.:
– Betreuen des Sachgebietes Computertechnik in der Zeitschrift
– Gewinnen und redaktionelles Bearbeiten von Manuskripten
– Besuchen und Auswerten von Fachtagungen, -messen und -ausstellungen
– Zusammenarbeit mit Gutachtern und ggf. selbständiges Testen von Programmen, die der Redaktion von Lesern zur Veröffentlichung eingereicht werden
– Bearbeiten bzw. Beantworten von Leserfragen.

Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, rufen Sie uns unter Tel. 2870203 oder 2870371 an, oder schreiben Sie an:

VEB Verlag Technik
Redaktion MP
Oranienburger Str. 13/14
Berlin
1020

Wir wünschen unseren Lesern und Autoren ein gesundes und erfolgreiches neues Jahr. Gleichzeitig hoffen wir, daß Sie uns auch 1988 bei der inhaltlichen Gestaltung und weiteren Profilierung unserer Zeitschrift unterstützen.

Ihre Redaktion MP

Moderne Kommunikationstechnologien

Prof. Dr. Dieter Hammer, Institut für Informatik und Rechentechnik der AdW der DDR
Prof. Dr. Dietmar Lochmann, Zentrum für Forschung und Technologie Nachrichtenelektronik

1. Einleitung

Das Informationsbedürfnis unserer Gesellschaft steigt gesetzmäßig mit der Zeit an. Neben dem Fernsprechen, das bisher den größten Teil des Informationsflusses im leitungsgebundenen Nachrichtennetz erzeugt, wächst das Bedürfnis an Text- und Datenübertragung sowie an Festbild- und Grafikübertragung, hervorgerufen durch die Bürokommunikation und die schnell fortschreitende Dezentralisierung der Rechentechnik bis hin zu jedem Arbeitsplatz.

Für die Volkswirtschaft der DDR wurden folgende Ziele zur Entwicklung moderner Kommunikationstechnologien gestellt /1/:

- Als Beitrag zum Erhöhen der volkswirtschaftlichen Produktivität und Effektivität sind durch den weiteren Auf- und Ausbau des automatisierten Daten- und Fernschreibnetzes die Möglichkeiten für einen schnellen Informations- und Datenaustausch zwischen den Kombinaten, Betrieben und staatlichen Einrichtungen zu erweitern sowie Voraussetzungen für das unmittelbare Zusammenwirken von EDV-Anlagen und Rechnernetzen zu schaffen.

- Auf dem Gebiet der elektronischen Nachrichtentechnik ist bis 1990 das Entwicklungstempo weiter zu erhöhen und der Übergang auf digitale Nachrichtentechnik für ausgewählte Geräte- und Softwarekomplexe zu vollziehen. Mit dieser Entwicklung ist der Prozeß der Integration bisher selbständiger und neuer Dienste, wie Sprachkommunikation, Fernkopieren, Fernschreiben, Daten- und Bildübertragung, zu einem System mit hohen volkswirtschaftlichen Effekten vorzubereiten.

Der Begriff der Kommunikationstechnologie setzt sich zusammen einerseits aus Kommunikation, d. h. der einseitigen oder wechselseitigen Abgabe, Übermittlung und der Aufnahme von Nachrichten durch den Menschen oder technische Systeme, und andererseits aus Technologie, die in diesem Zusammenhang sowohl als Methodik und Verfahren als auch als Synonym für Technik verstanden wird /2/. Darüber hinaus wird unter Kommunikationstechnologie die Telekommunikation impliziert, d. h. die Kommunikation mit Hilfe von Übertragungs- und Vermittlungsdiensten. Kommunikationstechnologien umfassen

– Netze für die Informationsübertragung
– Übertragungstechnik zur Steuerung der Übertragung auf Leitungen bzw. Kanälen des Netzes
– Vermittlungssysteme zur effektiven Nutzung der Netze für die Informationsübertragung
– Dienste, die auf der Grundlage der Eigenschaften der Vermittlungsprinzipien und Vermittlungstechnik dem Nutzer der Informationsübertragung angeboten werden
– Einrichtungen zur Nutzung der Kommunikationsdienste.

Die Entwicklung neuer hocheffektiver Kommunikationstechnologien ist mit einer immer stärkeren Annäherung von Nachrichten- und Rechentechnik verbunden, die insbesondere durch den zunehmenden Einsatz leistungsfähiger Rechnersysteme in Vermittlungssystemen und den umfassenden Einsatz der Mikroelektronik gekennzeichnet ist.
Sie erfordert die zielstrebige Entwicklung aller Teilgebiete, deren Entwicklung durch folgende Trends charakterisiert werden können:

Netze

– Erhöhen der Bandbreite/Übergang von Schmalband- zur Breitbandübertragung
– Einsatz von Lichtwellenleitern
– Satellitenübertragung unter Nutzung vorwiegend geostationärer Satelliten

Übertragungstechnik

– Übergang von der analogen zur digitalen Informationsübertragung
– Einsatz der Puls-Code-Modulation (PCM)
– Einsatz von Lichtwellenleiterübertragungssteuerungen

Vermittlungssysteme

– Übergang zur rechnergestützten Vermittlung einschließlich integrierter Formen der zeitgeteilten Digitaltechnik (PCM)
– Integration der Sprach- und Daten-/Textkommunikation im Rahmen eines Vermittlungssystems
– Integration aller Kommunikationsanforderungen einschließlich Bild- bzw. Bewegtbildkommunikation

Kommunikationsdienste

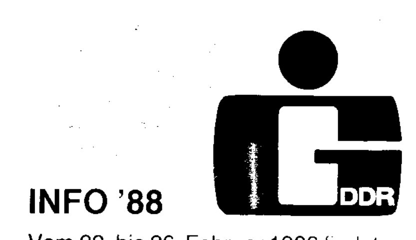
– Erweitern der einfachen Kommunikationsdienste (Fernsprechen, Telegrafie, Datenkommunikation) durch Informationsabruf-/-zugriffsdienste (Videotext, Teletex, Fernkopieren, Telekonferenz u. a.)
– Aufbau von Nachrichtenvermittlungssystemen (Message Handling Systems)
– Integration von Informationsverarbeitungsprozessen bei der Bereitstellung eines Dienstes

Nutzungseinrichtungen

Gewährleisten der Anschlußmöglichkeit an Datennetze für alle Rechner, insbesondere PC
– Bereitstellen von nutzerfreundlichen Endeinrichtungen für die anwendungsangepaßte Kommunikation (Schalterterminals, Bankterminals, öffentliche Videotext-Terminals. u. ä.)
– Bereitstellen von Endgeräten für neue Kommunikationsdienste und von Multifunktionsterminals durch das Ergänzen intelligenter Komponenten.

Moderne Kommunikationstechnologien sind erforderlich bzw. werden schrittweise entwikkelt und eingeführt, um das Informationsbedürfnis zur wirtschaftlichen und gesellschaftlichen Entwicklung in allen Ländern zu befriedigen. Untersuchungen des Internationalen Fernmeldevereins (ITU) weisen aus /3/, daß das Bedürfnis an Text- und Datenübertragung durch ein erheblich stärkeres Wachstum – etwa 20 % Zuwachs an Terminals pro Jahr – gegenüber einem Zuwachs von 4 % an Fernsprechstellen pro Jahr gekennzeichnet ist. In Bild 1 ist diese Entwicklung im Weltmaßstab dargestellt, wobei mehrere Datenkommunikationsmöglichkeiten unterschieden und insgesamt Trends ausgewiesen werden. Dabei entsprechen die unterschiedlichen Datenkommunikationsmöglichkeiten international standardisierten Diensten

– Telex für den seit vielen Jahren etablierten Fernschreibdienst
– Telefax für das Fernkopieren von Schriften und Grafiken
– Teletex für die Textkommunikation aus bis zu mehreren Seiten bestehenden Dokumentationen auf der Grundlage des Büroschreibmaschinenzeichenvorrats
– Videotext für die Nutzung des Fernsehbildschirms zur Ausgabe von Text auf dem Bild-



Vom 22. bis 26. Februar 1988 findet in Dresden der 4. Kongreß der Informatiker der DDR statt (siehe auch MP 12/87, 2. US).
Bei dem vorliegenden Artikel handelt es sich um den Vorabdruck eines Plenarvortrages der INFO '88. Wir beabsichtigen, im Laufe des Jahres weitere, ausgewählte Beiträge des Kongresses, die für unsere Leser von besonderem Interesse sind, zu veröffentlichen.

MP

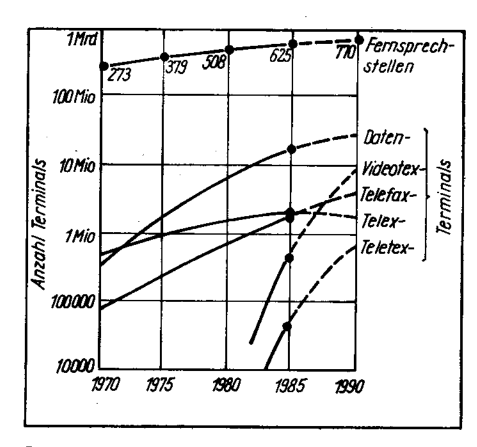


Bild 1 Entwicklung der Anwendung von Kommunikationsdiensten im Weltmaßstab

schirm bzw. „interaktiver Videotext" (Bildschirmtext) zur Nutzung des Fernsehgerätes in Verbindung mit dem Fernsprechnetz zur Auskunftsabfrage.
Nicht berücksichtigt ist bei dieser Betrachtung der erheblich höhere Informationsumfang der im Mittel von bzw. zu einer Datenstation gegenüber einer Fernsprechstelle übertragen wird.
Die Breite des Gegenstandes moderner Kommunikationstechnologien auf der Grundlage der Kommunikationsformen Sprache, Text, Bild und Daten (häufig unter Daten zusammengefaßt) erfordert eine Schwerpunktsetzung, da sie von der Datenfernübertragung über lokale und Weitverkehrsnetze, deren spezifischer technischer Realisierung und dem Ablauf der Kommunikationsprotokolle, von der Arbeitsweise in geschlossenen und offenen, insbesondere öffentlichen Nutzungssystemen, bis zu den über das Spezialgremium der ITU der CCITT regulierten Diensten, einschließlich der Fragen von Normen und Standards und vielen anderen Problemstellungen, reicht. Die weiteren Ausführungen konzentrieren sich daher auf einige aktuelle Fragen der Realisierung von Datenvermittlungssystemen für Weitverkehrsdatennetze und die Dienstintegration sowie auf langfristige Entwicklungstrends zur massenweisen Hochgeschwindigkeitskommunikation.

2. Datenvermittlungssysteme und Datennetze

Die Kommunikation in offenen Systemen im Sinne der Nutzung erfordert ein einheitliches Verständnis des Kommunikationssystems für jeden Nutzer unabhängig von der technischen Realisierung seiner Nutzungseinrichtung oder derjeniger, mit denen er eine Kommunikation wünscht. Dies führte zum Basis-Referenzmodell für die Kommunikation in offenen Systemen (kurz: OSI-Modell) /4/, das durch die Internationale Standardisierungsorganisation ISO als Standard erarbeitet und von der CCITT übernommen wurde. Die Basiselemente des Modells sind Anwendungsinstanzen als Abstraktionen von Anwendungsprozessen, die innerhalb von Systemen (Endsystemen) ablaufen, und Verbindungen, über die Informationen ausgetauscht werden (vgl. Bild 2). Über dem Übertragungsmedium werden 7 Funktionsschichten (Bitübertragungsschicht, Sicherungsschicht, Vermittlungsschicht, Transportschicht, Kommunikationssteuerungsschicht, Darstellungsschicht und Anwendungsschicht) festgelegt, die softwaremäßig über Prozeduren zu realisieren sind, sich über die einzelnen Systeme hinweg erstrecken und eine Hierarchie von Instanzen und Diensten für die jeweils darüberliegende Schicht bzw. den Anwenderprozeß erzeugen. Das OSI-Modell unterstützt die Einordnung und das Zusammenwirken von Nutzern über unterschiedliche Netze, mit verschiedenen Vermittlungssystemen sowie die Transparenz der Nutzung bei Veränderungen im Datennetz unter Beibehaltung des Nutzerdienstes und die internationale Datenkommunikation.
Die Datenvermittlungssysteme (DVS) bilden die technische Grundlage für die Datenübertragung zwischen Endsystemen, in denen die miteinander kommunizierenden Anwendungen als Nutzer lokalisiert sind. Netzseitig realisieren sie die Funktionen und Dienste entsprechend den Schichten 1 bis 3 des OSI-Modells. In Abhängigkeit von den Anwendungen entstehen unterschiedliche Anforderungen an DVS, wie z. B. bei der Stapelverarbeitung oder bei Echtzeitanwendungen oder bei Dialoganwendungen, was sich in differenzierten Anforderungen an

- das Zeitverhalten
- das Durchsatzverhalten
- die Dienstgüte (Vermittlungsgüte, Verkehrsgüte, Übertragungsgüte u. a.)

niederschlägt. Weitere Anforderungen des Nutzers betreffen Fragen der Sicherheit und Zuverlässigkeit der Datenübertragung oder auch die ökonomische Wirtschaftlichkeit. Entsprechend den Grundanforderungen an DVS haben sich für Stand- und Wählnetze die beiden Vermittlungsformen der Durchschaltevermittlung (Leitungsvermittlung) und der Paketvermittlung herausgebildet /5/. Bild 3 zeigt Formen der Datenvermittlung.
Bei der ***Durchschaltevermittlung*** wird der rufenden und der gerufenen Datenstation für die Dauer der Datenübertragung ein Übertragungsweg zur Verfügung gestellt. Nach dem Zustandekommen der Verbindung werden die Daten ohne Verzögerung oder Bearbeitung in den Datenvermittlungsstellen übertragen.
Bei der ***Paketvermittlung*** haben die Datenvermittlungsstellen Speicher zur Aufnahme der Daten. Sie haben die Funktion eines Knotens in einem Netz. Die in Paketen strukturierten Daten werden mittels Empfangen, Zwischenspeichern und Weitergeben von Knoten zu Knoten bis zum Zieladressaten entlang einem optimalen Weg durch das Netz gereicht.
Das Telefonnetz und das handvermittelte Datennetz unseres Landes funktionieren nach dem Prinzip der Durchschaltevermittlung. Das Kommunikationsnetz KOMET des

Dieter Hammer (49) schloß 1961 sein Mathematikstudium in Greifswald ab. Anschließend arbeitete er an der Entwicklung des ZRA1 im VEB Carl-Zeiss Jena. 1974 erfolgte die Promotion auf dem Gebiet der Betriebssysteme für Großrechenanlagen in der Akademie der Wissenschaften der DDR; 1981 die Promotion B auf dem Gebiet der Kommunikationsnetze auf Basis der Paketvermittlung. 1982 wurde er zum Professor für Informatik ernannt. Seit dieser Zeit ist Prof. Dr. Dieter Hammer Bereichsleiter im Institut für Informatik und Rechentechnik der AdW der DDR. Seine Arbeitsgebiete sind Rechnerarchitektur, fehlertolerante Rechnersysteme und Datennetze. Prof. Hammer ist Vorsitzender der Gesellschaft für Informatik der DDR.

Dieter Lochmann (55) studierte von 1953 bis 1958 an der TU Dresden Elektrotechnik. Anschließend nahm er eine Tätigkeit auf dem Gebiet der Kerntechnik auf. 1963 trat er als Assistent des Institutsdirektors in das Institut für Nachrichtentechnik ein. 1965 übernahm er die Funktion eines Bereichsleiters, und von 1970 bis 1986 war Prof. Dr. Dieter Lochmann als Direktor des Instituts tätig. Gegenwärtig ist er Direktor für Wissenschaft und Technik im Zentrum für Forschung und Technologie Nachrichtenelektronik. Seine wissenschaftlichen Arbeiten konzentrieren sich auf die Gebiete Datenübertragung und digitale Nachrichtentechnik. Schwerpunkte sind dabei z. Z. ISDN und digitale Signalverarbeitung.

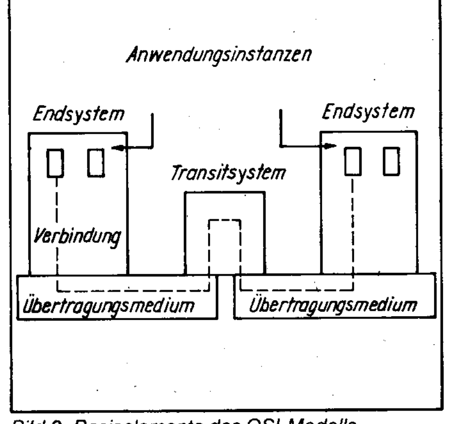


Bild 2 Basiselemente des OSI-Modells

Tafel 1 Gegenüberstellung Paketvermittlung und Durchschaltevermittlung

Paketvermittlung	Durchschaltevermittlung
Nur statistische Garantien für Übertragungszeiten, lastabhängig	Feste Übertragungszeiten
Übertragungskapazität nur im statistischen Mittel möglich	Übertragungskapazität pro Verbindung garantiert
Gegenseitige Beeinflussung von Datenströmen möglich	Keine gegenseitige Beeinflussung von Datenströmen im Netz
Zusätzliche Verwaltungsinformationen pro Paket erforderlich	Keine (oder weniger) redundante Daten
Komplexe Vermittlungsrechner-Logik	Einfache Vermittlungsrechner-Logik
Bei Belastung sinkt Datenübertragungsrate; ‚Anrufe' werden trotzdem angenommen;	Bei Belastung werden keine ‚Anrufe' mehr angenommen; einmal angenommene ‚Anrufe' werden mit der gleichen Übertragungsrate bearbeitet
Höhere Leitungsausnutzung im Netz	Geringe Leitungsausnutzung, vor allem bei interaktivem Datenverkehr
Pro Anschlußleitung max. 4095 Kommunikationen (logische Kanäle) möglich	Nur eine Kommunikation pro Anschlußleitung
Verbindungsaufbau in Bruchteilen von Sekunden	Verbindungsaufbau im Sek.-Bereich bis herab zu 400 ms
Tarif volumenabhängig, entfernungsunabhängig	Tarif zeitabhängig, dienstabhängig, entfernungsabhängig
Automatische Umsetzung der Übertragungsgeschwindigkeit zwischen verschiedenen Endeinrichtungen	Keine Umsetzung von Übertragungsgeschwindigkeiten zwischen verschiedenen Endeinrichtungen
Unterschiedliche Gerätetypen können miteinander kommunizieren	Nur kompatible Gerätetypen können miteinander kommunizieren
Bei Ausfall von Netzteilen (Knoten, Leitung) automatisches Umrouten möglich	Bei Ausfall, Verlust der Verbindung, kein automatisches Schalten von Ersatzleitungen

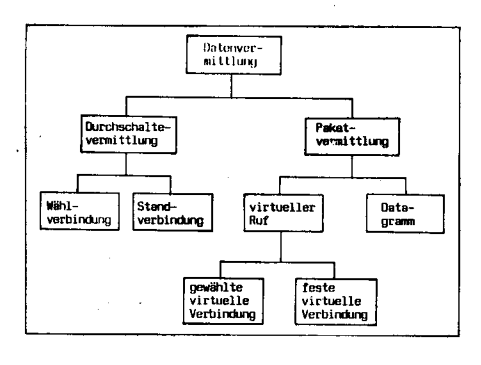


Bild 3 Formen der Datenvermittlung in Stand- und Wählnetzen

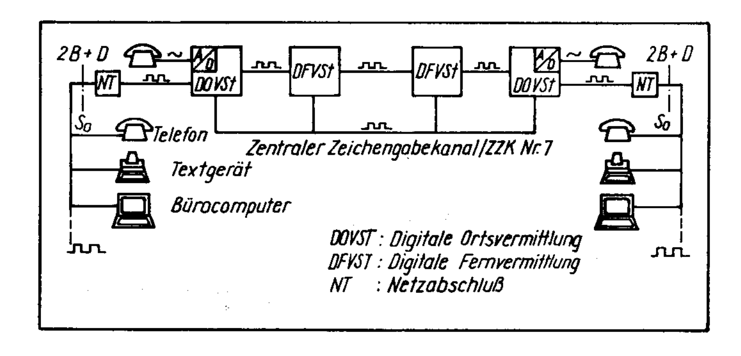


Bild 4 Struktur einer Verbindung im ISDN

Rechnernetzes DELTA für Forschung und Lehre sowie das experimentelle Datennetz der DDR EDN arbeiten nach dem Paketvermittlungsprinzip. International haben sich im Rahmen von öffentlichen Datennetzen Paketvermittlungsnetze in breitem Umfang, insbesondere zur Unterstützung der Datenkommunikation durchgesetzt. In Tafel 1 werden Paketvermittlung und Durchschaltevermittlung gegenübergestellt /2/. Schnittstellen, Dienste und viele weitere Parameter für Vermittlungssysteme sind von der CCITT (vor allem X-Serie) standardisiert, wobei der Standard X.25 an der Schnittstelle zwischen Datennetz und Endeinrichtung besonders hervorzuheben ist. Das Paketvermittlungsprinzip unterstützt vor allem eine dialogorientierte Kommunikation, was durch den ***Datagrammdienst*** – ein Dienst, der „Datagramme" (kurze Nachrichten) ohne Verbindungsauf- bzw. -abbau auf Grund der im Kopf des Datagramms enthaltenen Adresse durch das Datennetz zur Zielendeinrichtung überträgt – besonders unterstützt wird. Für längere Nachrichten wird der Dienst des virtuellen Rufs (***virtual call***) genutzt, der mit dem Aufbau einer virtuellen Verbindung zwischen rufender und gerufener Station für die Dauer der Übertragung einer Nachricht sich logisch an das Prinzip der Durchschaltevermittlung anlehnt, jedoch auf Grund der Vermittlung aller Pakete von Knoten zu Knoten die Leitungskapazität nur während der Übertragung der Daten belastet, d. h., eine Leitung kann gleichzeitig vielen virtuellen Verbindungen zugeordnet sein, virtuelle Verbindungen können zeitweilig (gewählt) oder permanent aufgebaut werden (vgl. Bild 3).

Ein wesentliches Ziel der Weiterentwicklung von Vermittlungssystemen besteht darin, die Vorteile beider Vermittlungsformen optimal für spezifische Nutzeranforderungen zu nutzen.

3. Anforderungen und Lösungen für Paketvermittlungsnetze

Die zunehmende Bedeutung der nationalen und internationalen Datenkommunikation für weite Bereiche der Volkswirtschaft führte zum Aufbau bereichsspezifischer und öffentlicher Paketvermittlungsnetze in vielen Ländern. Dabei sind Anforderungen zu erfüllen, die den Anschluß von 10000en (gegenwärtig) bis zu 100000en Leitungen (zukünftig) – bei hoher Zuverlässigkeit im Sinne der Gewährleistung der Verfügbarkeit der Datenübertragungsdienste – ermöglichen. Diese Anforderungen sind durch die rechentechnischen Systeme in den Knoten des Netzes und in den Kontroll- und Netzverwaltungszentren zu erfüllen. Nach /6, 7/ können sie durch folgende Parameter charakterisiert werden:

- bis zu 10000 Leitungsanschlüsse pro Knoten
- Durchsatz: 20000 bis 30000 Pakete/s, 600 bis 1000 Rufe/s (Anforderungen zum Aufbau einer virtuellen Verbindung)
- Verfügbarkeit: > 99,997 %
- flexible und modulare Struktur
- kostengünstige Realisierung kleiner, mittlerer und großer Datenübertragungssysteme
- Netzanschlüsse mit einer Übertragungsrate von 2 Mbit/s.

Zur Realisierung der Paketvermittlung sind für jedes Paket in einem Knoten folgende Aufgaben zu erledigen:

- Empfang über eine Leitung
- Überprüfen der Richtigkeit der Übertragung und übertragenen Daten, sonst Aufforderung des Senders zur Wiederholung des Übertragens
- Aufbau/Abbau oder Zuordnung zu einer virtuellen Verbindung
- Feststellen des weiteren Weges durch das Netz bei Rufen
- Übertragen zum nächsten Knoten entsprechend dem vorbestimmten Weg oder zu einer Endeinrichtung und erforderlichenfalls Wiederholen bei nicht korrekter Übertragung

Zusätzlich sind viele Funktionen zu realisieren, die den Datentransport nicht unmittelbar betreffen, wie

- nutzerbezogenes Erfassen der Übertragungsleistung
- Erfassen von Fehlern und Anzeigen
- Bereitstellen von Ausweichlösungen
- Unterstützen von Tests
- Erweitern oder Einschränken des Netzes durch Zu- bzw. Abschalten von Leitungen oder Knoten
- Verändern von Steuerungsparametern u. a.

Derartige Aufgaben sind sowohl in den Knotenrechnersystemen als auch in den Kontroll- und Verwaltungssystemen zu realisieren. Alle Aufgaben sind bei einer ununterbrochenen Arbeit des Netzes zu sichern. Um die hohen Leistungs- und Zuverlässigkeitsanforderungen an die Echtzeitverarbeitungssysteme in den Knoten zu erfüllen, können keine universellen Standardrechnersysteme eingesetzt werden.

Fehlertolerante Multiprozessorsysteme, die über Hochgeschwindigkeitsverbindungen zu Mehrrechnerkomplexen verbunden werden, gestatten die flexible Gestaltung leistungsabgestufter Systeme, in denen durch dynamische Redundanz die Verfügbarkeit bei Ausfällen gewährleistet wird.

Die geforderten Zuverlässigkeitsparameter können nur durch den gemeinsamen Einsatz von Fehlervermeidungs- und Fehlertoleranzmaßnahmen erreicht werden. Die funktionsbeteiligte Redundanz von Hardware-Modulen und die Möglichkeiten der on-line-Reparatur sind zu ergänzen durch Mittel zur schrittweisen Fehlererkennung bzw. -maskierung, Fehlerbegrenzung, Fehlerdiagnose, Rekonfiguration, Wiederanlauf- und Reintegration. Zwischen den Prozessormoduln des komplexen Rechnersystems in einem Knoten muß eine Kommunikation zum Austausch von Informationen für die Paketvermittlung zur Organisation der verteilten Arbeit des Betriebssystems und zur Realisierung von Fehlertoleranzmaßnahmen des Betriebssystems gewährleistet sein. Das Betriebssystem für ein solches System muß echtzeitfähig sein und als verteiltes System auf allen Prozessormoduln wirken. Wichtige Aufgaben bestehen darin:

- durch die Hardware angezeigte bzw. aus eigenen Kontrollen festgestellte Fehler zu erkennen und zu diagnostizieren
- die Fehlerwirkung einzugrenzen und beeinflußte Systemteile aus der Arbeit herauszulösen
- über einen Rekonfigurationsmechanismus die Fortführung derartiger Prozesse auf den redundanten Ressourcen zu organisieren und
- die Integration reparierter Ressourcen (damit auch von Ressourcen zur Erweiterung) zu unterstützen.

Darüber hinaus ist die Software für die Prozesse der Paketvermittlung für ein solches Multiprozessor-Mehrrechnersystem zu programmieren und zu testen. Das erfordert geeignete Compiler für die von der CCITT für derartige Software geforderte Programmiersprache CHILL bereitzustellen. Der Test der Programme und die Erprobung des gesamten Softwaresystems setzt den Zugriff zu einer Vielzahl von Werkzeugen der Softwaretechnologie voraus.

International gibt es mit den Systemen 1PSS (Philips, AT&T) /8/, DPN (Northern Telecom) /9/ und EWSP (Siemens) /7, 10/ Entwicklungen zu Knotenrechnersystemen für Paketvermittlungsnetze, die die obengenannten Parameter anstreben bzw. erreichen. Alle drei Systeme weisen zur Realisierung der Paketvermittlungsfunktion unterschiedliche Systemarchitekturen auf. Ungeachtet dessen werden in allen Systemen fehlertolerante Multiprozessorsysteme eingesetzt, die auf unterschiedliche Weise zu einem Rechnerkomplex miteinander verkoppelt sind, um die Leistungsziele zu erreichen. Zur Lösung spezieller Teilaufgaben werden teilweise in bereits seit mehreren Jahren in Datennetzen er-

probte Kleinrechnersysteme zur Unterstützung der Kompatibilität integriert.
Die breite Palette an Anforderungen an die Rechentechnik unterstreicht einerseits das immer engere Zusammenwachsen von Rechentechnik und Nachrichtentechnik, andererseits stimuliert sie die Forschung und Entwicklung von zuverlässigen, hochverfügbaren Rechnersystemen mit hoher Leistungsfähigkeit bei der Bearbeitung paralleler Prozesse für volkswirtschaftliche Aufgaben auch völlig anderer Anwendungsrichtungen.

4. Dienstintegration

Daten- und Fernsprechnetze haben unterschiedliche Anforderungen an das Übertragungs- und Vermittlungssystem. Dennoch werden Fernsprechnetze auf Grund ihrer großen Ausdehnung und territorialen Verfügbarkeit international gegenwärtig etwa zu 30% für Daten- und Textübertragungsdienste genutzt /3/. Trotz der Begrenzung des Fernsprechkanals auf 3,4 kHz werden heute durch Anwendung moderner Verfahren der digitalen Signalverarbeitung Datenübertragungsgeschwindigkeiten im vermittelten Fernsprechnetz bis 9,6 kbit/s erreicht. Aus dem raschen Wachstum der Anforderungen an die Datenübertragung resultiert das Bestreben, die Unterschiede zwischen Daten- und Fernsprechübertragung abzubauen und wie beim Telefonanschluß z. B. jeden Büroarbeitsplatz auch mit einem Datenanschluß zu versehen. Als entscheidender Schritt hierzu ist die Digitalisierung des Fernsprechnetzes anzusehen. Auf der Basis eines digitalen Fernsprechnetzes kann die Integration aller Dienste in einem einzigen Netz – dem Integrated Services Digital Network (ISDN) – realisiert werden. Ein ISDN wird charakterisiert durch

- gleichzeitige digitale bittransparente Mehrfachkommunikation mit zwei Basiskanälen B zu je 64 kbit/s vollduplex von Teilnehmer zu Teilnehmer
- Anschluß der Endgeräte (Terminals) über eine Zweidrahtkupferleitung
- Anwendung eines separaten Signalisierungskanals D mit 16 kbit/s zum Teilnehmer
- Nutzung einer einheitlichen Geräteschnittstelle S_0 mit 2B + D = 144 kbit/s
- Anwendung digitaler Einrichtungen für die Vermittlung und Übertragung im gesamten Netz
- Anwendung des zentralen Zeichengabekanals CCITT-Nr. 7 zwischen den Vermittlungsstellen
- Nutzung aller Schmalbanddienste für Sprache, Text, Daten, Grafik über eine Informationssteckdose und eine Rufnummer.

Im Bild 4 wird die Struktur eines ISDN dargestellt. Danach sind von Teilnehmer zu Teilnehmer unter einer Rufnummer zwei bittransparente digitale 64 kbit/s-Kanäle durchschaltbar. Diese Kanäle arbeiten vollkommen unabhängig voneinander, d. h. sie können für verschiedene Dienste, z. B. Fernsprechen und Festbildübertragung, oder zu verschiedenen Teilnehmern benutzt werden. Mit der Digitalisierung bis zum Teilnehmer wird die Einrichtung einer Einheits-Kommunikationssteckdose möglich, über die alle genannten Dienste durch den Einsatz entsprechender Endgeräte abgewickelt werden können. Der separate Signalisierungskanal zum Teilnehmer wird im ISDN als D-Kanal bezeichnet. Dieser Kanal endet meist in der digitalen Ortsvermittlungsstelle, an die der Teilnehmer angeschlossen ist. Er überträgt alle Signalisierungsinformationen für die beiden Basiskanäle, z. B. Selektionskennzeichen und Rufnummern. Der D-Kanal arbeitet paketorientiert in Anlehnung an die Datenübertragung im paketvermittelten Datennetz. Die Ablaufsteuerung erfolgt nach dem D-Kanal-Protokoll. Im Gegensatz zum D-Kanal sind die beiden B-Kanäle leitungsvermittelt, d. h., ihre Verbindung zum anderen Teilnehmer bleibt während der Übertragungs- bzw. Gesprächszeit elektrisch durchgeschaltet. Wie aus Bild 4 ersichtlich ist, kann man an eine ISDN-Vermittlung auch „Nur-Fernsprechteilnehmer" anschalten. Für ISDN sind bereits von der CCITT etwa 30 Standardisierungsempfehlungen erarbeitet worden, während weitere gegenwärtig vorbereitet werden. Die Zweckmäßigkeit von ISDN wird dadurch bestimmt, daß Datenanschlüsse auf der Grundlage des existierenden Fernsprechnetzes an jeden Arbeitsplatz gebracht werden können. Die ökonomischen Vorteile ergeben sich aus folgenden Betrachtungen:

- In den dichtbesiedelten Industrieländern Europas liegt die durchschnittliche Länge einer Fernsprechanschlußleitung bei 2 bis 2,5 km, die eines Datenanschlusses bei über 50 km. Der Datenanschluß ist also wesentlich aufwendiger und kostenintensiver.
- Im Ortskabelnetz stecken schon heute über 40% der Investaufwendungen für ein Fernsprechnetz. Die Fernsprechanschlußleitungen sind bereits verlegt und bisher schlecht ausgenutzt.
- Schätzungen besagen, daß international die Anzahl der an ein Netz anzuschließenden Daten- und Textgeräte (CAD-Stationen) im Vergleich zum Fernsprechen von 7% (1982) auf 50% (1995) ansteigen wird /6/.

Weitere große Vorteile besitzt ein ISDN im Hinblick auf die Universalität des Anschlusses mit bis zu 8 Endgeräten an einheitlichen Steckdosen, auf komfortable Leistungen beim Fernsprechdienst und auf hohe Übertragungsgeschwindigkeiten bei Datendiensten.
Von besonderer Bedeutung sind Möglichkeiten der Paketvermittlung im ISDN. Der Zugang des ISDN zum getrennten Paketvermittlungsnetz erfordert einen X.25-Protokoll-Adapter für die Schnittstelle S_0. Für die weltweit stark ansteigenden dialogorientierten Datendienste haben Paketvermittlungsnetze gegenüber ISDN weiterhin betriebliche und wirtschaftliche Vorteile /3/, was dazu führt, daß Paketvermittlungssysteme hoher Leistung weiterentwickelt und ausgebaut werden. Langfristig wird als ein Ziel verfolgt, die Paketvermittlung voll in das ISDN zu integrieren.
Dazu ist vorgesehen, für den Aufbau und Abbau von virtuellen Verbindungen das D-Kanalprotokoll anstelle des X.25-Protokolls zu nutzen. Gegenwärtig wird in der CCITT an entsprechenden Standardisierungsempfehlungen für diesen „New Packet Mode" gearbeitet.

5. Entwicklungstrends

Die technischen Voraussetzungen für die Anwendung von Lichtwellenleitern in Weitverkehrsnetzen sind gegeben, so daß sich optische Nachrichtenübertragung von digitalen Informationen auf Kanälen mit hoher Bandbreite vollziehen kann. Auf Grund der technisch-betrieblichen und wirtschaftlichen Vorteile der Lichtwellenleitertechnik wird dies langfristig über das Jahr 2000 hinaus eine entscheidende Entwicklung darstellen. Dabei ist davon auszugehen, daß Lichtwellenleiter bis zum Teilnehmeranschluß zum Einsatz gelangen. Durch die damit verfügbaren höheren Bandbreiten kann die digitale Bewegtbildübertragung neben allen anderen bereits genannten Kommunikationsdiensten integriert werden. So können zum einen ISDN-Kanäle mit Bitraten von 2 bis 140 Mbit/s zum ***Breitband-ISDN*** homogen weiterentwickelt werden.

Zum anderen kann auf Grundlage der Fortschritte der Mikroelektronik bei der Entwicklung von Bauelementen mit hohen Schaltgeschwindigkeiten die Paketvermittlung auch im Breitbandbereich angewendet werden, wobei ein besonderer Vorteil im Zusammenfassen (Multiplexen) unterschiedlicher Bitraten einzelner Kommunikationsdienste besteht. Zu zwei bemerkenswerten Entwicklungsrichtungen auf dem Gebiet der Breitbandpaketvermittlung werden gegenwärtig intensive Forschungsarbeiten durchgeführt:

- ***Burst- oder Broadcast Packet Switching*** (BPS)
- ***Fast Packet Switching*** (FPS).

Sie stellen gegenüber dem Breitband-ISDN einen anderen Weg für eine Universallösung dar und bilden eine Alternative zum Breitband-ISDN /13/. Burst-Packet-Switching-Netze bestehen aus Paketvermittlungseinrichtungen (PVE) als Knoten und Netzinterfaceeinrichtungen, über die Nutzer mit dem Netz verbunden werden können. Die Netzinterfaceeinrichtungen realisieren Konzentrator-Schutz- und Abrechnungsfunktionen. Sie können einerseits mit bis zu 300 Nutzern über Lichtwellenleiter und andererseits über ein Bündel von Lichtwellenleitern (bis zu 30) mit einer PVE verbunden sein. Eine PVE ist wiederum über Bündel von Lichtwellenleitern mit anderen PVE verbunden. An einer PVE können 60 bis 900 Netzinterfaceeinrichtungen angeschlossen werden, d. h., über eine PVE können bis zu 27000 Lichtwellenleiter zur Herstellung von Verbindungen zu 200000 Nutzern angeschlossen werden. Als Dienste werden Punkt-zu-Punktkanäle (Nutzer-Nutzerkanäle), die Datagrammübertragung und Broadcast-Kanäle, d. h. von einem Nutzer zu vielen Nutzern, angeboten. Damit können alle bekannten Kommunikationsdienste einschließlich von Bewegtbildübertragung (Fernsehen) oder auch Telekonferenzen auf Sprach- bzw. Bildbasis durch Paketvermittlung realisiert werden. Ein derartiges System soll nicht nur industrielle und Bürokommunikation befriedigen, sondern den Anforderungen des massenhaften Anschlusses auch von Haushalten gerecht werden.

Diese gewaltige Leistung wird durch hochparallele fehlertolerante Rechnerarchitekturen erreicht.

– Eine PVE setzt sich aus bis zu 500 gleichartigen Vermittlungsmoduln zusammen.
– Ein Vermittlungsmodul besteht aus einer sogenannten „Vermittlungsfabrik", an der bis zu 63 E/A-Paketprozessoren und ein Steuerprozessor angeschlossen sind. Alle Komponenten sind aus Zuverlässigkeitsgründen im Sinne einer statischen Redundanz gedoppelt.
– Jeweils ein E/A-Paketprozessor steuert den Empfang oder das Senden von Paketen auf einem Lichtwellenleiter. Beim Empfang werden Weginformationen (Routing) und Broadcast-Informationen aus den Paketen entnommen und dem Steuerprozessor übergeben. Empfangene Pakete werden an die Vermittlungsfabrik weitergereicht.
– Die Vermittlungsfabrik besteht aus Netzverbindungsstrukturen von 64 × 64 binären Mehrstufennetzen, bei denen die Schaltfunktion in den Knoten für jedes Paket durch Adressen aus den Routing- bzw. Broadcast-Informationen über den Steuerprozessor gestellt wird.

Auf diese Weise wird ein nahezu verzögerungsfreier Fluß der Datenpakete realisiert. Eine Paketvermittlungseinrichtung gewährleistet einen Durchsatz von über $5 * 10^9$ bit/s, was mehr als 1,2 Mio Paketen/s mit einer Paketlänge von maximal 594 Bytes entspricht. Das Fast Packet Switching ist ähnlich dem Burst Packet Switching auf die Vermittlung von Paketströmen für eine große Nutzeranzahl unter Einsatz von Breitbandkanälen orientiert. Das Konzept basiert auf festen logischen Kanälen, die in einer Baumstruktur von der Quelle aus andere Nutzer erreichen. Die hohe Übertragungsgeschwindigkeit im Netz wird durch eine nahezu auf Null reduzierte Vermittlungsverzögerung in den Transitknoten für den Datentransport bei virtuellen Verbindungen erzielt.

Literatur

/1/ Dokumente des XI. Parteitages der SED, Direktive zum Fünfjahrplan für die Entwicklung der Volkswirtschaft der DDR in den Jahren 1986 bis 1990. Dietz Verlag Berlin 1986, S. 64–65, 89–90
/2/ Goergen, K.; Koch, H.; Schulze, G.; Strui, B.; Truoel, K.: Grundlagen der Kommunikationstechnologie, Springer-Verlag, 1985, S. 1
/3/ Wiest, G.: Öffentliche Kommunikationsnetze – Evolution der Technik und Veränderung der Märkte, 1987
/4/ ISO/DIS7942 Draft International Standard, ISO/TC97/SC5/WGN163, 1983
/5/ Hammer, D.: Kommunikationsnetze unter Nutzung der Paketvermittlung. ZfR-Informationen, ZfR-I-80.03, 1980
/6/ Runkel, D.: Datex-P, The Public Packet Switching Network of the Deutsche Bundespost after five Years of Experience. Proc. of ICCC 86, München
/7/ Mair, W.; Hansmann, H.; Naessel, R.: EWSP-A High Performance Packet Switching System. Proc. of ICCC 86, München
/8/ Denssing, P.; Eckhardt, B.: The Distributed IPSS Architecture. A High Reliable Switch for High-Performance Packet Switching Network. Proc. of Intern. Conference on Computer Communication. München, Sept. 1986
/9/ Wile, G. E.; Gowan, D. S.: The Architecture of the DPN Date Networking System. Proc. of ICCC 86, München, S. 365 ff.
/10/ Huber, J. F.; Mair, E.: Universelle Paketvermittlung durch flexible EWSP-Architektur. telecom report 10 (1987) 1
/11/ Geirhofer, H.: Der Übergang vom analogen zum digitalen Fernsprechnetz und weiter zum ISDN. Postrundschau (1985) 1, S. 20–23, Österreich
/12/ Bocker, P.: Das Kommunikationsnetz der Zukunft: Konzept und Anwendungen. NTF, Band 88, S. 28–41
/13/ O'Reilly, P.: Burst and Fast-Packet Switching: Performance Comparisons. Proc. of IEEE INFOCOM '86, 1986, S. 653–666
/14/ Turner, J. S.: Design of a Broadcast Packet Network. s. /13/, S. 667–675
/15/ Saadawi, T.; Jain, N.; Schwartz, M.: Protocol for a Distributed Switching Interactive CATV Network, s. /13/ S. 676–684

Zur Weiterentwicklung bipolarer Bauelemente

Prof. Dr. Johannes Godau
VEB Halbleiterwerk Frankfurt (Oder)

1. Vorbemerkungen

Bis 1990 ist das in der DDR vorhandene Schaltkreissortiment durch die Einführung neuer Basistechnologien zur Beherrschung wesentlich verringerter Strukturbreiten und die Realisierung spezifischer komplexer Informationsverarbeitungsfunktionen mittels höchstintegrierter Schaltkreise zu erweitern.
Für den VEB Halbleiterwerk Frankfurt (Oder) (VEB HFO) leitet sich daraus die Verantwortung für die Bedarfsdeckung folgender Schaltkreissortimente ab:
– digitale IS für die Datenverarbeitung
– analoge und digitale IS für die industrielle Elektronik
– analoge IS für die Konsumgüterelektronik
– Wandler- und Präzisionsbauelemente
– IS für die Nachrichtentechnik
– kundenspezifische Schaltkreise nach dem HFO-ISA-System.

Die Realisierung dieser Sortimente erfolgt mittels speziell entwickelter Bipolar- und Mischtechnologien.
Aus den Anwenderforderungen ergibt sich die Notwendigkeit der Entwicklung skalierter Bipolar- und Mischtechnologien mit höheren technologischen Niveaus gegenüber dem jetzigen Stand im VEB HFO, um der Anwenderindustrie auch in Zukunft moderne und dem internationalen Stand entsprechende Bauelemente bereitstellen zu können.

2. Internationale Entwicklung und Hauptrichtungen der Bipolar- und Mischtechnik

Die Bipolartechnik hatte 1986 einen Anteil von ca. 40 % am Bauelementeweltmarkt, die restlichen ca. 60 % wurden durch unipolare Bauelemente abgedeckt. Durch das weitere Vordringen der MOS-Technik, insbesondere der CMOS-Technik, in klassische Einsatzgebiete bipolarer Bauelemente, wie den Logiksektor, wird sich der Anteil der Bipolartechnik am Bauelementemarkt bis 1990 voraussichtlich auf 33 % reduzieren, wobei sich absolut gesehen trotzdem ein Wachstum ergibt.
Die Reduzierung des Anteils der Bipolartechnik am Weltmarkt ist bedingt durch die Vorteile, die sich für die MOS-Techniken im Ergebnis umfangreicher F/E-Arbeiten herauskristallisiert haben. In vielen Fällen hat sich aber auch aus diesen F/E-Arbeiten ergeben, daß die Unipolartechnik nicht die Werte der Bipolartechnik mit dem Vorteil der Leistungsarmut erbringen kann.
Die für den Aufbau neuer F/E- und Produktionskapazitäten in der DDR getroffenen Entscheidungen berücksichtigen diese Tendenz. Der VEB HFO als der traditionelle Hersteller bipolarer Bauelemente in der DDR muß aus diesem Entwicklungstrend unter Berücksichtigung der spezifischen Bedingungen und Anforderungen, die sich aus der Gesamtsituation in der Volkswirtschaft ergeben, entsprechende Schlußfolgerungen für die Profilierung unserer Bipolartechnik ziehen. Diese werden auf der Grundlage der materiell-technischen Basis der Mikroelektronik der DDR und der Bedarfsentwicklung in der Grundrichtung mit dem internationalen Entwicklungstrend übereinstimmen.

2.1. Logik-Schaltkreise für die Computertechnik

Logik-Bauelemente machen derzeit fast die Hälfte der Bipolarbauelemente aus. Gefertigt werden sie insbesondere in Schottky- bzw. Low-Power-Schottky-TTL-Technik, aber auch in TTL- und ECL-Technik. Aufgrund der Vorteile der CMOS- gegenüber den TTL-Techniken bezüglich Stromaufnahme, Breite des Betriebsspannungsbereiches, Betriebstemperatur und Störfestigkeit wird bereits jetzt ein Teil der Logik-Bauelemente in CMOS-Technik hergestellt.
Dieser Anteil wird sich bei weiteren Fortschritten der CMOS-Technik bezüglich der Schaltgeschwindigkeit weiter erhöhen. Mit der HC/HCT-Reihe hat die CMOS-Technik inzwischen Schaltgeschwindigkeiten erreicht, die mit denen der LS-TTL vergleichbar sind, die neue AHCT-Reihe kommt in den Bereich der ALS-TTL.
In den nächsten Jahren wird es aufgrund der spezifischen Vorteile der CMOS- und der Bipolartechnik auf dem Logik-Sektor zu einer Aufteilung nach Anwendungsbereichen kommen:
• Für schnelle und sehr schnelle Logik, insbesondere im SSI- und MSI-Bereich, wird vorrangig die Bipolartechnik eingesetzt werden. Innerhalb dieser Gruppe wird es eine Verschiebung zugunsten der ECL geben.
Die dabei zu realisierenden Schaltzeiten liegen im Bereich 1 bis 0,1 ns pro Gatter.
• Alle anderen Einsatzfälle, für die die Geschwindigkeit nicht im Vordergrund steht, werden durch CMOS-Logik-Bauelemente abgedeckt.

Der Zielbereich für moderne CMOS-Technologien im Technologieniveau 1–2 μm liegt bei den Schaltzeiten zwischen 10 und 1 ns pro Gatter.
Der Bedarf an schnellen und sehr schnellen Logik-Bauelementen wird sich in den nächsten Jahren weiter erhöhen, die Geschwindigkeitsanforderungen steigen im Zusammenhang mit den Entwicklungen zum Beispiel in der Rechentechnik und der Lichtleiterübertragungstechnik.
Die Bipolartechnik hat in diesem Bereich folgende Vorteile:

- Sie weist bei gleichem Technologieniveau um den Faktor 5–10 kleinere Schaltzeiten auf, bzw. bei gleichen Schaltzeitforderungen gestattet sie den Einsatz eines Technologieniveaus, das um zwei Stufen unter dem erforderlichen Technologieniveau der CMOS-Technologie liegt.
- Für die Ausgangsstufen und damit für die Anpassung an die Peripherie besitzt sie höhere Treiberleistungen und niedrigere Ausgangswiderstände.
- Das Geschwindigkeits-Leistungsprodukt wird um den Faktor 10 reduziert (bei 25 MHz).
- Die dynamische Verlustleistung ist geringer, damit erfolgt ein geringerer Anstieg der Verlustleistung bei steigender Arbeitsfrequenz.
- Sie weist keinen Latch-up-Effekt auf (Durchbrennen der Bauelemente bei Betriebsspannungsspitzen).
- Sie erfordert keine gesonderten Schutzmaßnahmen an den Eingangs- und Ausgangsstufen gegen Störspannungsspitzen von den Verbindungsleitungen.

Schnelle und sehr schnelle Logik-Schaltkreise werden benötigt für die Gebiete

- 32-Bit-Mikroprozessor-Systeme
- PCM-Technik
- Rechentechnik
- digitale Vermittlungssysteme
- Lichtleiternachrichtenübertragung
- Meßtechnik.

Dabei steht neben der Forderung nach höheren Verarbeitungsgeschwindigkeiten auch die Forderung nach komplexeren Schaltkreisen, das heißt die Realisierung höherer Integrationsgrade. Damit ist die Entwicklung in Richtung hoher Verarbeitungsgeschwindigkeit und Komplexität auf dem Gebiet der bipolaren Logik-Schaltkreise ziemlich eindeutig definiert.

2.2. *Wandlerschaltkreise*

Wandlerschaltkreise sind Interface-Schaltkreise zwischen der analogen Umwelt und der digitalen Signalverarbeitung. Ausgehend von den vielseitigen Anwendungsgebieten dieser IS in der Meß-, Steuerungs- und Regelungstechnik entstehen die unterschiedlichsten Forderungen bezüglich Auflösung, Wandlungsgeschwindigkeit und funktioneller Komplexität. Aus diesem Grunde kommt es bei der Auswahl der zu realisierenden Gesamtfunktion der Wandler darauf an, einen universellen Einsatz zu gewährleisten, um die Typenvielfalt dieser komplizierten Präzisionsschaltkreise zu begrenzen, damit eine möglichst kostengünstige Produktion möglich wird.
Mit dem vorhandenen bzw. in Entwicklung befindlichen Wandlersortiment wird z. Z. den wesentlichsten Anforderungen der Gerätehersteller entsprochen. Zu diesem Sortiment gehören die integrierenden A/D-Wandler mit einer Auflösung und Genauigkeit von 3 bis 4 1/2 digit, die schnellen A/D-Wandler mit einer Auflösung und Genauigkeit von 8 bis 12 Bit und die entsprechenden D/A-Wandler der gleichen Genauigkeitsklasse.
Dieses Wandlersortiment ist pinkompatibel mit international eingesetzten Industriestandard-Schaltkreisen und bestimmt den internationalen Stand mit.
Die internationale Entwicklung zeigt folgenden Trend:
Ausgehend von der Tatsache, daß die Parameter der Wandler maßgeblich das Einsatzgebiet der digitalen Signalverarbeitung bestimmen, sind weltweit Arbeiten zur Verbesserung der Wandlungsgeschwindigkeit, Auflösung und funktionellen Komplexität im Gange. Dieser Prozeß läßt sich in folgende Entwicklungsrichtungen untergliedern:
1. Ultraschnelle Wandler für die Videotechnik, Bilderkennung, Spektralanalyse und Computergrafik. Dafür werden international bereits A/D- und D/A-Wandler mit einer Auflösung bis zu 8 Bit und einer Wandlungsgeschwindigkeit bis zu 100 MHz angeboten.
2. Hochauflösende Wandler mit einer Auflösung und Genauigkeit von 12 bis 16 Bit für den Einsatz in der Meß- und Steuerungstechnik sowie für die Audiotechnik.
3. Erhöhung der Komplexität durch Mikroprozessor-Kompatibilität und durch die Integration von zwei bis vier Wandlern auf einem Chip.

Zur Realisierung dieser drei Entwicklungsrichtungen werden Spitzentechnologien eingesetzt, die folgende Parameter aufweisen:

- Für die ultraschnellen Wandler ist die Bipolar-Technologie mit Strukturabmessungen von 1 bis 2 μm dominierend.
- Für hochauflösende Wandler werden Mischtechnologien (Bipolar-Unipolar) eingesetzt.
- Zur Erhöhung der Komplexität der Wandler-IS und für Wandler, die nicht die Forderungen der ersten beiden Punkte erfüllen müssen, werden in breiter Form MOS-Technologien mit zwei bis drei Leitbahnebenen und minimalen Strukturabmessungen von 2 μm angewandt.

Zusammenfassend läßt sich sagen, daß international das Lithografieniveau mit 2-μm-Strukturen dem Stand der Technik entspricht und eine Anpassung der Technologien an die Anforderungen der Wandler entsprechend ihrer Einsatzgebiete stattfindet.
Seitens der Anwenderindustrie werden Wandler für die digitale Bildverarbeitung (schnelle 8-Bit-A/D- und D/A-Wandler) und hochauflösende 16-Bit-Präzisions- und Audio-Wandler gefordert. Des weiteren gehen die Forderungen in Richtung der Verbesserung der Mikroprozessor-Kompatibilität und der Auslegung der Wandler für nur eine Betriebsspannung (5 V). Im VEB HFO besteht die Aufgabe, auf der Grundlage des perspektivischen strategischen Bauelementesortimentes des VEB Kombinat Mikroelektronik das Sortiment der Wandler- und Präzisionsschaltkreise weiterzuentwickeln.

2.3. *Kundenspezifische Schaltkreise*

International hat in den letzten Jahren die Herstellung und Anwendung von anwendungsspezifischen integrierten Schaltkreisen (ASIC) stark an Bedeutung gewonnen. Je nach Einsatzgebiet und benötigter Stückzahl gibt es neben den üblichen Vollkunden-IS nachfolgende Gruppen von ASICs:

- Standardzellen-IS
- Gate-Arrays
- Programmierbare Logik-IS (PLA)
- Programmierbare Festwertspeicher (PROM).

Allen Gruppen ist ein bestimmter Vorfertigungsstand gemeinsam, die Unterschiede liegen im Entwurfssystem bzw. im Fertigungsablauf.
Bei Standardzellen-IS müssen, wie bei Vollkunden-IS, alle Maskenebenen anwendungsspezifisch erstellt werden. Dazu sind die vorentworfenen elektrischen Teilfunktionen als sogenannte Makros in Bibliotheken gespeichert. Die bereits flächenoptimierten und in ihren Daten verbindlich beschriebenen Makros werden zum Gesamtschaltkreis zusammengesetzt.
Die Gate-Arrays, PLAs und PROMs sind in großen Stückzahlen vorgefertigt und erhalten ihre anwenderspezifische Funktion durch die entsprechenden Verdrahtungsmasken in den letzten Teilschritten der Scheibenherstellung bzw. durch Programmierung. Durch die Anwendung von ASICs ergeben sich generell zwei wesentliche Vorteile:

- Auch kleine Stückzahlen sind ökonomisch realisierbar.
- Eine erhebliche Zeiteinsparung bis zur Bereitstellung und Testung erster Muster gegenüber einem Vollkunden-IS.

Um diese Vorteile voll zu nutzen, sind jedoch zu Beginn der Entwicklung eines kundenspezifischen IS folgende Bedingungen zu prüfen bzw. zu erfüllen:
1. Das Projekt muß für eine Integration geeignet sein, daß heißt, die Wirtschaftlichkeit ist zu überprüfen.
2. Die Design-Schnittstellen müssen klar definiert werden, das heißt Festlegung der Arbeitsteilung zwischen Anwender und Hersteller für die einzelnen Entwurfsschritte.

Prof. Dr. rer. nat. Johannes Godau (48) studierte von 1957 bis 1963 am Leningrader Elektrotechnischen Institut „Uljanow/Lenin" in der Fachrichtung Halbleiter/Dielektrika. Anschließend am gleichen Institut bis 1966 Aspirantur auf dem Gebiet Physikalische Chemie von Halbleiterverbindungen.
Seit 1966 ist Prof. Dr. Godau im VEB Halbleiterwerk Frankfurt (O.) tätig – zunächst als Laborleiter, Auftragsleiter und 1. Stellvertreter des Direktors für Forschung und Entwicklung, von 1973 bis 1977 als Direktor für Beschaffung und Absatz.
Seit 1978 ist Prof. Dr. Godau Direktor für Forschung und Technologie.
1986 wurde er von der Akademie der Wissenschaften der DDR zum Professor ernannt; 1979 Auszeichnung mit dem Nationalpreis der DDR für Wissenschaft und Technik.

3. Der Schaltungsentwurf und der IS-Prüfentwurf (Meßtechnik) müssen abgeschlossen sein. Nachträgliche Modifikationen erhöhen den Zeit- und Kostenaufwand erheblich.

Die Erfahrungen bei der Anwendung des HFO-ISA-Systems zeigen ganz deutlich, daß gerade die Beachtung dieser Bedingungen mit einem Lerneffekt behaftet ist. Das heißt, erst wenn es der Anwender gelernt hat, mit einem ASIC-IS-System umzugehen, werden dessen Vorteile voll wirksam.
In der DDR wird der Einsatz von kundenspezifischen Schaltkreisen als ein Weg zur flexiblen Erweiterung der IS-Sortimente eingeschätzt. Der VEB HFO ist seit 1982 Entwickler und Produzent bipolarer Array-Schaltkreise. Insgesamt stehen 11 Arrays im MSI- und LSI-Niveau zur Verfügung, auf denen analoge, digitale und gemischte Schaltungen entworfen werden können.
Steigende Gatter- und Strukturzahlen bei den Arrays und das Bestreben, den Entwicklungszeitaufwand von kundenspezifischen Schaltkreisen zu verringern, machen computergestützte Entwurfsmittel, also CAD-Systeme, für den Anwender unentbehrlich. Diesem Trend folgend, bietet der VEB HFO zu zwei neuentwickelten Arrays eine komplette IS-Entwurfssoftware (ISACAD) an.
Das Programmsystem ISACAD setzt die anwendungsspezifische Schaltung weitgehend automatisch in das Verdrahtungslayout um. Eine ISACAD-Version für komplexe analoge Schaltungen wird in Zusammenarbeit mit der TU Dresden seit Jahresbeginn auf Basis von zwei weiteren Grundchips bereitgestellt.
Für ISA-Arrays wurden im VEB HFO im Produktionsprozeß befindliche Technologien eingesetzt. Dieses Prinzip garantiert die für vorproduzierte Arrays notwendige technologische Stabilität und ist auch Voraussetzung für Array-Neuentwicklungen. Die Weiterentwicklung des Systems maskenprogrammierbarer bipolarer Arrays bis 1990 ist im VEB HFO auf der Grundlage der zu entwickelnden neuen Technologien konzipiert. Das Prinzip des durchgängigen automatischen Entwurfs ist dabei u. a. auf zwei Verdrahtungsebenen auszubauen. Diese Aufgabe wird in enger Forschungskooperation mit der TU Dresden bearbeitet und gesichert.

3. Produktionsentwicklung und Bedarfsdeckung

Die Produktion bipolarer Schaltkreise soll, bezogen auf das Jahr 1986, im VEB HFO bis zum Jahre 1990 verdoppelt werden.
Der Bedarf der Hauptanwenderkombinate umfaßt folgende Anteile des Inlandbedarfs (in %):

Robotron	46
Nachrichtenelektronik	8
Automatisierungsanlagenbau	4
Mikroelektronik	6
Elektro-Apparate-Werke Berlin	10
Rundfunk und Fernsehen	13

Einen maßgeblichen Einfluß auf den Prozeß der Bedarfsdeckung hat die Außenwirtschaftstätigkeit mit dem Schwerpunkt der Importsicherung zur Komplettierung des eigenen Sortiments.
Die große Einsatzbreite der Mikroelektronik in der DDR erfordert die Bereitstellung einer großen Typenvielfalt auf dem Gebiet der mikroelektronischen Bauelemente. Zur Sicherung der Ökonomie ist deshalb eine zielgerichtete Spezialisierung im RGW und entsprechender Bauelementeimport notwendig. Betrachtet man das Jahr 1986, so betrug der Anteil des Importes, gemessen an der wertmäßigen Warenproduktion des VEB HFO auf der Basis IAP, rund 60 %; das heißt, mehr als die Hälfte des Eigenaufkommens in der Erzeugnisposition bipolare Halbleiterbauelemente wird gegenwärtig importiert.
Der hohe Bedarf unserer Anwender zwingt zu neuen Überlegungen im Rahmen der Außenwirtschaftstätigkeit. Die bestehende Ministervereinbarung mit der Republik Kuba sowie Verhandlungen mit der UVR und der SFRJ zum Auf- und Ausbau von Kooperationsbeziehungen führen zu zusätzlichen Importmöglichkeiten. Der Schwerpunkt der Spezialisierung im Import, das heißt die langfristig vertraglich vereinbarte Lieferzusage, umfaßt folgende Sortimente:

– Ergänzungstypen der Standard-TTL-Schaltkreise
– Schottky-TTL-Schaltkreise
– ECL-Schaltkreise
– Interface-Schaltkreise
– Festspannungsregler und Spannungsstabilisatoren
– Si-Kleinleistungs- und Leistungstransistoren für NF- und HF-Anwendungen.

Eine wichtige Aufgabe beim weiteren Ausbau unserer Produktionskapazitäten bildet die Sicherung des Erneuerungsgrades unserer Produktion. Es ist deshalb eine konsequente Substitution veralteter Bauelemente in der Anwenderindustrie durchzusetzen. Dabei ist auch der Import von langfristig im RGW im Angebot befindlichen Bauelementen zu nutzen.

4. Zusammenarbeit mit der Anwenderindustrie

Die Applikationstätigkeit im VEB HFO ist die Grundlage einer intensiven und vertrauensvollen Zusammenarbeit mit der Anwenderindustrie. Sie wird von den Direktoraten Forschung und Technologie und Absatz arbeitsteilig vorgenommen, wobei sich die Struktureinheiten des Direktorats Absatz insbesondere auf die Aufgaben der Marktforschung und Sortimentsstruktur in Abhängigkeit von vorhandenen Importmöglichkeiten sowie auf Mustervergaben (auch Entwicklungsmuster) und Versand von Informationsmaterial konzentrieren. Die Hauptarbeitsrichtungen der Applikationsarbeit sind

- Bauelementestrategie
- applikative Betreuung in Produktion befindlicher Bauelemente
- Öffentlichkeitsarbeit.

Die Zielstellung der Arbeiten zur Bauelementestrategie beinhaltet die Erarbeitung eines perspektivischen Sortimentes, das sowohl die technischen Möglichkeiten des VEB HFO als auch die berechtigten Forderungen der Anwenderindustrie sowie die Verpflichtungen aus der Außenwirtschaft berücksichtigt.
Es kommt darauf an, die eigenen Vorstellungen zur Sortimentsstrategie unter Berücksichtigung internationaler Trends mit den wichtigsten Forderungen der Anwenderindustrie, die in der Regel aus der Untersetzung zentraler Vorgaben resultieren, zu vergleichen und konkrete Entwicklungszielstellungen festzulegen.
Diese Gesamtproblematik kann nur in enger Zusammenarbeit mit den Anwendern gelöst werden und setzt voraus, daß die Geräteindustrie weiß, was sie in den nächsten Jahren unbedingt braucht.
Bei monovalent einsetzbaren Bauelementen ist das Problem in der Regel gut lösbar, da man es mit einem Partner zu tun hat und dieser die Priorität seiner Forderungen mit vorlegt.
Wesentlich komplizierter gestaltet sich die Abstimmung bei multivalent einsetzbaren Bauelementen, für deren Ausführung es verschiedene Alternativen gibt oder ganze Baureihen inhaltlich festgelegt werden müssen. Für diese Arbeiten steht uns als nationales Gremium die Sektion Schaltungsintegration mit ihren Fachgruppen zur Verfügung. In diesem Rahmen kommt es zu regelmäßigen Abstimmungen zwischen den Vertretern der Bauelementeindustrie und den wichtigsten Anwenderkombinaten. Daher ist die Arbeit in der Sektion Schaltungsintegration für die gesamte Applikationstätigkeit unseres Betriebes von großer Bedeutung.
Zu den wichtigsten Ergebnissen der Sektionsarbeit gehören die Festlegungen zu den Logikreihen (Arbeitsteilung HFO/MME), die Operationsverstärker- und Wandlerstrategie sowie die Festlegungen der IS für die Schreib- und Drucktechnik.
Im Rahmen unserer weiteren Applikationstätigkeit werden wir verstärkt den Kontakt mit den Anwendern suchen, um in größerem Maßstab Anfalltypen zu verkaufen. Dies liegt im Interesse der Materialökonomie und hilft bei Bedarfsdeckungsproblemen. Es ist nicht zu vertreten, daß Bauelemente, die in vielen Fällen voll einsetzbar sind, nicht verwertet werden. Dabei muß in jedem Fall gesichert sein, daß die technische Qualität des Finalproduktes nicht gefährdet wird.
Zur Bewältigung dieser Aufgabe erwarten wir die konstruktive Mitarbeit der Geräteindustrie im Interesse des beiderseitigen Vorteils. Als weiteren Schwerpunkt der Applikationstätigkeit werden wir in der nächsten Zeit alle Anstrengungen unternehmen, den Auslauf der Produktion veralteter Bauelemente zu erreichen, um die Produktionskapazitäten für moderne Erzeugnisse zu nutzen. Die Bereinigung unseres Typensortimentes in diesem Sinne liegt sicherlich auch im Interesse der Geräteindustrie und wir erwarten deshalb ihre Unterstützung.
Besonders schwierig gestaltet sich die Ablösung der Schaltkreise für die industrielle Elektronik, insbesondere der Standard-TTL-Reihen, obwohl nahezu alle logischen Funktionen in der Low-power-Schottky-Technik zur Verfügung stehen. Die Lösung dieses Problems wird den Schwerpunkt unserer Bemühungen bilden. Im Interesse dieser notwendigen, gemeinsamen Anstrengungen werden wir alle Möglichkeiten nutzen, wobei die Substitution durch neue Bauelemente den Vorrang haben muß.

✉ KONTAKT ☏

VEB Halbleiterwerk Frankfurt (O.), Postfach 379, Frankfurt (Oder), 1200; Tel. 46 23 92 (Koll. Walther)

Mikroprozessorgesteuertes Positioniersystem mit 16-Bit-CPU

Prof. Dr. Werner Liebich,
Dr. Michael Krapp,
Technische Hochschule Ilmenau,
Sektion Technische und Biomedizinische Kybernetik
Reinhard Langmann, VEB Robotron-Rationalisierung Weimar

1. Vorbemerkungen

Internationale Vergleiche zeigen, daß die Anforderungen an die Präzision und Dynamik des Bewegungsverhaltens moderner Montageroboter in der Feingeräte- und Elektroindustrie nur durch den Einsatz mikrorechnergesteuerter Positioniersteuerungen für die einzelnen Achsantriebe erfüllt werden können. Dabei gewinnt die Implementierung leistungsfähiger digitaler Regelalgorithmen, die das erforderliche Folgeverhalten des Reglers sowie Zeitoptimalität und Überschwingfreiheit der Bewegung sichern, zunehmend an Bedeutung.
Der Aufbau hochautomatisierter Arbeitszellen unter Beteiligung von Industrierobotern und peripheren Antriebseinheiten erfordert weiterhin aufgrund der räumlichen Trennung der Komponenten serielle Verbindungssysteme (lokale Netzwerke), die einen schnellen Informationsaustausch zwischen Steuerrechner und dem jeweiligen Achsrechner (Positioniersteuerung) ermöglichen.
Darüber hinaus verlangt die Vielzahl der durch Positioniersteuerung im Echtzeitbetrieb zu lösenden Aufgaben gute Dokumentierbarkeit und Flexibilität für die Anpassung an verschiedene Einsatzfälle.
Eine kompakte und ökonomische Realisierung der genannten Forderungen an eine Positioniersteuerung ermöglicht erst die fortgeschrittene LSI-Technik durch den Aufbau strukturierter Mehrprozessorsysteme mit hoher Rechenleistung /1, 2/. Im folgenden wird ein solches Prozessorsystem auf der Grundlage des 8-Bit-Einchipmikrorechners (EMR) UB 8820 und der 16-Bit-CPU UB 8002 vorgestellt.

2. Hardware

Die Probleme des Hardwareentwurfs für das Positioniersystem bestanden vor allem darin, die parallele funktionelle Struktur einer Positioniersteuerung in die parallele technische Struktur eines lokalen Mehrprozessorsystems optimal umzusetzen. Ausgehend vom erforderlichen Funktionsumfang bilden drei Mikroprozessoren gemeinsam mit zugeordneten digitalen und analogen Baugruppen – bezeichnet als Controller-Funktionseinheiten – ein Drei-Prozessor-System, bei dem der Informationsaustausch über einen gemeinsamen Bus und einen Koppelspeicher erfolgt.
Die drei Funktionseinheiten Netzwerkcontroller, Servocontroller und Positionscontroller (Bild 1) realisieren die Aufgaben

– Anschluß an ein für Echtzeitsteuerung geeignetes lokales Datennetz
– Feininterpolation zwischen Bahnstützpunkten und Abarbeitung von Lageregelalgorithmen
– Aufbereitung inkrementeller Winkelmeßsignale zu einer totalen Istposition und Anpassung an den eingesetzten elektrischen Antrieb.

Bild 2 zeigt das Prinzipschaltbild des Positioniersystems. Als Prozessorbauelemente kommen der EMR UB 8820 und die 16-Bit-CPU UB 8002 zum Einsatz. Die 16-Bit-CPU sichert vor allem die schnelle Abarbeitung rechenintensiver Algorithmen. Zur Kommunikation über den Koppelspeicher erfolgt eine dezentrale Busvermittlung des gemeinsamen Busses im Zeitmultiplexbetrieb. Der EMR 1 startet den Zugriff auf den Koppelspeicher im Taktraster 1,5 ms. Das Zugriffssignal wird als ZS 0 und ZS 1 wie beim Daisy-Chain-Verfahren von Prozessor zu Prozessor durchgereicht (realisiert über Software) und gestattet damit den Zugriff des jeweiligen Prozessors auf den gemeinsamen Speicher. Das Zugriffsverfahren ist schaltungstechnisch sehr einfach zu realisieren und für den vorliegenden Fall einer festen Aufgabenverteilung auf die einzelnen Prozessoren völlig ausreichend.
Die beiden EMR sind über PORT 1 direkt gekoppelt, da dieses Tor softwaremäßig als Interface für externe Speicher betrieben werden kann. Zwischen Adreßdatenbus der beiden EMR und dem gemeinsamen Bus befindet sich ein Bustreiber. Sowohl die Steuerung dieses Bustreibers als auch die R/W-Steuerung des Koppelspeichers erfolgen über Adreßsignale vom PORT 0 sowie die Speichersteuersignale $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ des jeweiligen EMR (in Bild 2 nicht gezeichnet).
Bestandteil des Netzwerkcontrollers ist ein aus den Bustreiber-/-empfänger-Schaltkreisen DL 2631/32 aufgebauter Transceiver, der die unsymmetrischen seriellen Ein-/Ausgangsleitungen des EMR 1 an eine symmetrische Zwei-Draht-Leitung (lokales Datennetz) anpaßt. Da über softwaremäßige Mittel gesichert ist, daß an dieser Zwei-Draht-Leitung immer nur ein Sender arbeitet, können mehrere Positioniersysteme räumlich entfernt an diese Leitung angeschlossen werden.

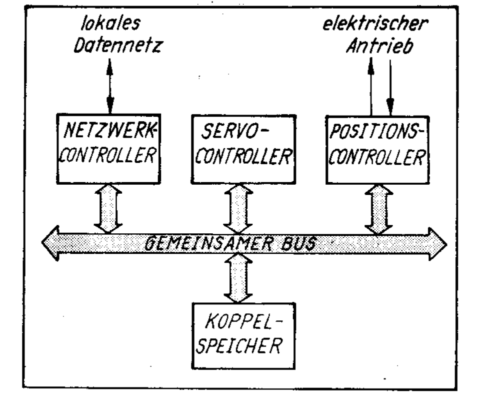


Bild 1
Funktionseinheiten des Positioniersystems

Der Positioniercontroller beinhaltet neben dem EMR 2 eine schnelle Hardwarestruktur (Antriebsinterface) mit folgenden Aufgaben:
– Verarbeitung der inkrementellen Winkelmeßsignale Z1, Z2, N eines IGR und Bildung eines zyklischen Istpositionsvektors
– D/A-Wandlung der Stellgröße I 1 (Strom) für Gleichstromantriebe mit mechanischem Kommutator
– D/A-Wandlung und Erzeugung der Statorstromkomponenten I 1, I 2 im Statorkoordinatensystem für die Steuerung von Drehfeldantrieben (Synchron-, Asynchronmotoren).

Eine ausführliche Beschreibung dieses Antriebsinterfaces erfolgte in /3, 4, 5/. Der Aufbau ist durch digitale Automaten mit Speicherschaltkreisen U 2716 und multiplizierenden D/A-Wandlern DAC 7520 realisiert.
Ein U-8000-Minimalsystem gewährleistet die erforderliche Rechenleistung. Das Lagereglerprogramm wird hardwaremäßig durch das Signal ZS 1 am NMI-Eingang gestartet. Die Kommunikation erfolgt über die untere Hälfte eines Wortes (Byteadressierung).
Gemeinsam mit einer WATCH-DOG-Schaltung, die die Funktion der drei Prozessoren überwacht, erfordert die gesamte Hardware ca. 65 Schaltkreise und läßt sich auf einer Leiterplatte (215 × 170) mm in Mehrebenentechnik aufbauen.

3. Software

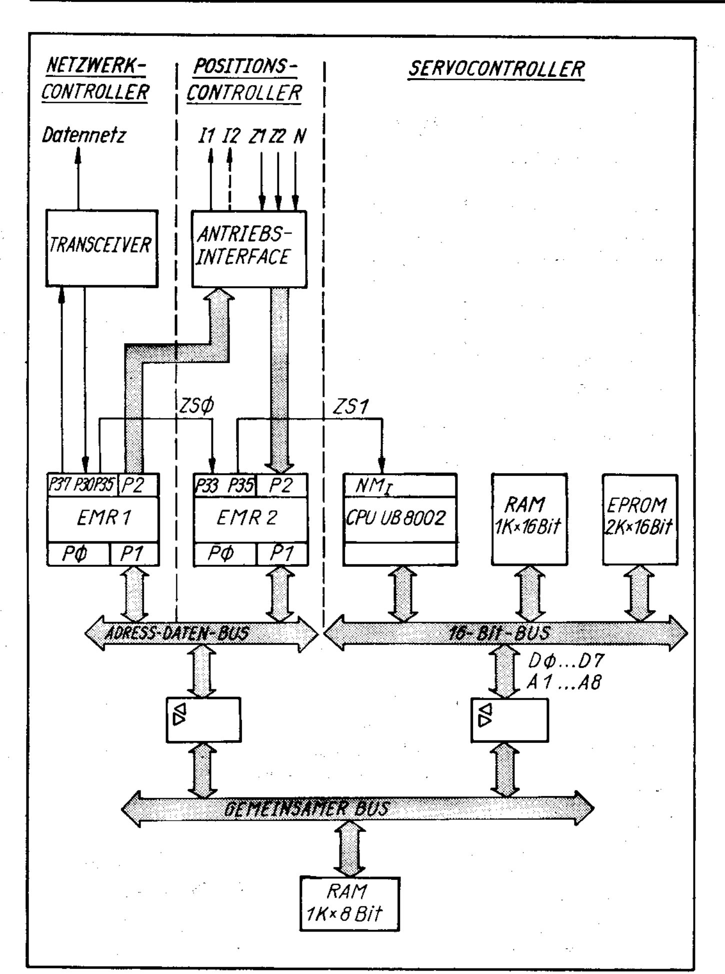
Aufgrund des strukturierten Aufbaus des Positioniersystems kann die Software getrennt für jeden Prozessor unter Beachtung der Übergabebedingungen im Koppelspeicher entwickelt werden. Sie ist damit leicht austauschbar und einfach an die spezifischen Aufgaben des entsprechenden Controllers anpaßbar.

3.1. Positionscontroller

Die Hauptaufgabe für das zugehörige EMR-Programm besteht in der laufenden Berechnung der totalen Istposition und der Kommunikation mit dem Koppelspeicher.
Aus dem zyklischen Zählvektor ermittelt das Programm vorzeichenrichtig den Winkelzuwachs am Rotor des Stellmotors pro Zeiteinheit und addiert diese Werte zu einer 32-Bit-Istposition auf. Dieser Programmteil arbeitet interruptgesteuert.
Das maximal mögliche Taktintervall wird durch die erforderliche Auflösung des IGR, die Drehzahl des Motors und die Breite des zyklischen Zählvektors begrenzt.
Zur Reduzierung der Laufzeit des Programms erfolgte eine solche Optimierung, daß möglichst nur Operationen mit Registern der gleichen Registergruppe durchgeführt werden. Die Bestimmung der totalen Istposition aus einem 7-Bit-Zählvektor erfordert damit maximal 80 μs.

Dazu addieren sich noch Programmlaufzeiten für die
– Überwachung der Maximalgeschwindigkeit (26 μs)
– Kontrolle einer Referenzposition (40 μs)
– Fehlerüberwachung der Zähllogik (15 μs).

Das Zugriffssignal ZS 0 löst eine höherpriori-



Prof. Dr. sc. techn. Werner Liebich (52) studierte von 1954 bis 1960 an der Martin-Luther-Universität Halle und an der Technischen Hochschule (jetzt Technische Universität) Dresden. 1967 Promotion A und 1976 Promotion B am Institut für Kybernetik Kiew.
Von 1967 bis 1975 war er als Abteilungsleiter im Zentrum für Forschung und Technik des VEB Kombinat Robotron tätig. 1975 erfolgte die Berufung zum Ordentlichen Professor für Gerätetechnik der Informationsverarbeitung an die Sektion Technische und Biomedizinische Kybernetik der Technischen Hochschule Ilmenau. 1977 bis 1982 Prorektor für Erziehung und Ausbildung, Stellvertreter des Sektionsdirektors für Forschung, Leiter des Wissenschaftsbereiches Technische Informatik.

Doz. Dr. sc. techn. Michael Krapp (42) studierte von 1963 bis 1969 an der Technischen Hochschule Ilmenau; 1974 Promotion A, 1981 Promotion B an der Technischen Hochschule Ilmenau.
Von 1969 bis 1979 war er als Assistent und Oberassistent auf dem Gebiet der Informationsverarbeitung an der Sektion Technische und biomedizinische Kybernetik der Technischen Hochschule Ilmenau tätig. Danach arbeitete er im VEB Kombinat Robotron als Abteilungsleiter an der Entwicklung von Robotersteuerungen. 1986 erfolgte die Berufung zum Hochschuldozenten für digitale Automaten und formale Sprachen an die Sektion Technische und Biomedizinische Kybernetik der Technischen Hochschule Ilmenau, wo er am Wissenschaftsbereich Technische Informatik tätig ist.

Dipl.-Ing. Reinhard Langmann (37) studierte am Leningrader Elektrotechnischen Institut, Fachrichtung Halbleiterelektronik.
Nach einer Tätigkeit auf dem Gebiet des wissenschaftlich-technischen Rechtsschutzes für Erfindungen beschäftigt er sich seit 1980 im VEB Robotron-Rationalisierung Weimar mit der Entwicklung von Industrierobotersteuerungen. Sein Spezialgebiet sind Hard- und Softwareprobleme der Bewegungssteuerung von Präzisionsrobotern.
1987 erfolgte die Einreichung seiner Dissertation A an der Technischen Hochschule Ilmenau zum Entwurf mikroprozessorgesteuerter Positionierantriebe für die Präzisionsrobotertechnik.

Bild 2
Prinzipschaltbild des Positioniersystems

sierte Interruptserviceroutine aus, die in den Koppelspeicher eine Statusinformation sowie aktuelle Istposition und -geschwindigkeit einschreibt und ein Betriebsartsteuerwort sowie eine Korrekturposition ausliest. Durch die zeitaufwendige Art der externen Speicheradressierung des UB 8820 werden für diese Kommunikation gleichfalls 80 μs benötigt.
Mit der maximalen Programmlaufdauer von

$$T_\Sigma = T_{Pos} + T_{KOMM} \approx 250\,\mu s$$

konnte am Laboraufbau die totale Istposition am Rotor des Stellmotors bei einer Auflösung $p \approx 0{,}09°$ bis zu Drehzahlen $n = 2300\ min^{-1}$ fehlerfrei ermittelt werden.
Zur Ansteuerung von Drehfeldantrieben müssen vom EMR 2 zusätzliche Steuerparameter für die Feldorientierung an das Antriebsinterface ausgegeben werden (über P3 bzw. P2 im Multiplexbetrieb). Der Zeitbedarf für die Bildung dieser Parameter richtet sich nach dem Kompliziertheitsgrad des gewählten Motormodells. Die Abarbeitung zeitaufwendiger Algorithmen kann dazu auch im Servocontroller erfolgen.

3.2. Servocontroller

Im Servocontroller erfolgt die Lageregelung. Als Testvariante wurde ein Zustandsregler mit integraler Ausgangsrückführung gemäß

$$U(K) = U(K-1) + R1 \cdot e(K) + R2 \cdot e(K-1) + R3 \cdot e(K-2)$$

U(K) – Stellgröße im K-ten Taktintervall
e(K) – Winkeldifferenz im K-ten Taktintervall
R1, R2, R3 – Reglerparameter

implementiert.

Die hohe Rechenleistung der 16-Bit-CPU, insbesondere für erforderliche Arithmetikoperationen mit doppelter Wortbreite, gestattet eine Berechnung der Stellgröße bei veränderbaren Reglerparametern während der Bewegung des Stellmotors. Damit bestehen die rechentechnischen Voraussetzungen, um adaptive Lageregler einzusetzen.
Zur Berücksichtigung stochastischer Lastschwankungen an der Motorwelle wird gegenwärtig ein selbsteinstellender Regler mit Adaptivsteuerung der Parameter untersucht. Das Lagereglerprogramm stellt praktisch die NMI-Routine des 16-Bit-Minimalrechners dar. Nach dem Start des Programms über ZS 1 erfolgt zuerst Lesen/Schreiben des Koppelspeichers und danach die Abarbeitung des Lageregelalgorithmus. Im Hintergrundprogramm findet eine Plausibilitätskontrolle aller Informationen im Koppelspeicher statt.

3.3. Netzwerkcontroller

Der Netzwerkcontroller schafft die Verbindung zu einem geeigneten lokalen Datennetz, bezeichnet als Motor-Control-Bus (MC-Bus). Das zugehörige EMR-Programm bearbeitet folgende Hauptaufgaben:

– Kommunikation mit dem Koppelspeicher
– Anschluß an das Datennetz
– Adressierung für Sende- und Empfangsaufruf
– Codeumwandlung
– Nachrichtensteuerung und -vermittlung.

Der Zugang zum MC-Bus wird kollisionsfrei über das Control-Token-Verfahren realisiert, das heißt, die Weitergabe der Netzwerkkontrolle erfolgt von Teilnehmer zu Teilnehmer.
Die Einsatzspezifik des Positioniersystems (Steuerung von Industrierobotern) erfordert vom MC-Bus
a) eine Versorgung der angeschlossenen Teilnehmer mit Steuerdaten im Zeitraster < 20 ms und
b) die On-line-Programmierung der Teilnehmer bezüglich ihrer Betriebsart und ihrer Netzwerksteuerung.
Die Erfüllung der Forderung a) erweist sich mit dem EMR aufgrund der relativ niedrigen Bitrate seiner seriellen Schnittstelle (max. 62,5 Kbit/s) als problematisch. Verwendet man jedoch für die Steuerung der angeschlossenen Antriebsmodule nur relative Positionsdaten, können über den MC-Bus sechs Achsen in ca. 8 ms bedient werden.
Zur Realisierung der Forderung b) werden neben Steuerdaten auch Steuerbefehle übertragen, die einerseits die Betriebsart von Positions- und Servocontroller festlegen und andererseits die Art und Weise der Nachrichtensteuerung und -vermittlung durch den Netzwerkcontroller bestimmen.

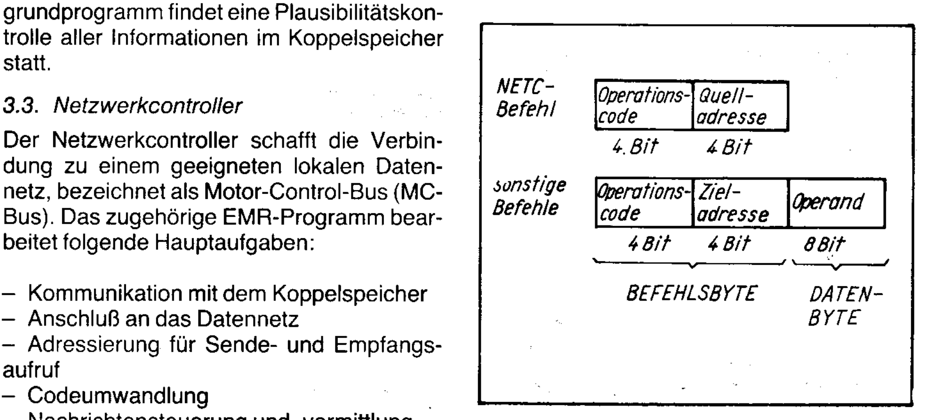


Bild 3
Befehlswortformat für den Nachrichtenaustausch über den MC-Bus

Tafel 1 Befehlsliste für das Positioniersystem, die der Netzwerkcontroller verarbeitet.

STEUERBEFEHLE

Mnemonik	Befehlsfunktion
NETC	Netzwerkkontrollbefehl (Token)
PROG	Programmierbefehl
MODE	Betriebsart des Servocontrollers
PARA	Betriebsart des Servocontrollers
CNTR	Status- und Kontrollbefehl

DATENMANIPULATIONSBEFEHLE

Mnemonik	Befehlsfunktion
SPOA	Sollposition 1. Byte
SPOB	Sollposition 2. Byte
SPOC	Sollposition 3. Byte
IPOA	Istposition 1. Byte
IPOB	Istposition 2. Byte
IPOC	Istposition 3. Byte
SOLV	Sollgeschwindigkeit
SOMA	Sollmoment 1. Byte
SOMB	Sollmoment 2. Byte

Für den Nachrichtenaustausch ist das in Bild 3 dargestellte Befehlswortformat festgelegt. Der NETC-Befehl besteht nur aus einem Steuerbyte und wirkt als Control-Token.
Zur Sicherung der Kompatibilität zu Standardschnittstellen wird jedes Befehls- bzw. Datenbyte in Form von zwei Zeichen im ASCII-Code übertragen.
Tafel 1 gibt eine Übersicht über die durch den Netzwerkcontroller verarbeitbaren Steuer- und Datenmanipulationsbefehle. Mit dem Befehl PROG wird das nachfolgende Datenbyte als Programminformation interpretiert und in einem Programmspeicher abgelegt.
Bild 4 verdeutlicht den Wirkmechanismus des Netzwerkcontrollers für die Betriebsarten
- Senden (durchgezogene Verbindung)
- Empfang von Befehlen/Daten (gestrichelte Verbindung)
- Programmieren (Strich-Punkt-Verbindung).

Durch die gegenüber kommerziellen Systemen vereinfachte Netzwerkarchitektur kann über den MC-Bus eine effektive Datenrate von 35 Kbit/s erreicht werden, wobei bis zu 15 Teilnehmer am Bus angeschlossen werden können.
Zusätzlich zu den Aufgaben der Netzwerkkontrolle und der Kommunikation mit dem Koppelspeicher übernimmt der EMR 1 aus hardwaretechnischen Gründen (Ausnutzung der IN/OUT-Tore) die Ausgabe der Stellgröße für den Antrieb (s. Bild 2).

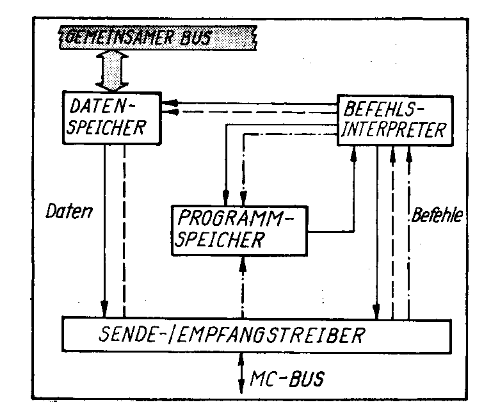


Bild 4
Wirkmechanismus des Netzwerkcontrollers

4. Schlußbemerkung

Am Labormuster konnte das Positioniersystem erfolgreich getestet werden. Als Antriebe wurden sowohl ein Gleichstromkollektormotor (GMP 52) wie auch ein als momentgestellter Synchronmotor betriebener Schrittmotor (DW 250–3000) mit gechopperten Leistungsverstärkern eingesetzt /4/. Die Messung der Istposition an der Rotorwelle erfolgte durch einen IGR–M2/960 mit 2- bzw. 4fach-Interpolation.
Mit dem vorgestellten Positioniersystem konnte der Nachweis erbracht werden, daß auch ohne den Einsatz von Spezialprozessoren (Signalprozessor, Arithmetikprozessor), durch Kombination von Standardprozessortechnik mit geeigneten digitalen und analogen Schaltmitteln ein kompakter Aufbau leistungsfähiger Positioniersysteme möglich ist.
Das Positioniersystem ist für den Einsatz in Nachfolgeentwicklungen des sensorgeführten Montageroboters P 1050 vorgesehen.

Literatur

/1/ Bodenkamp, J.; Bitbus: Verteilte Intelligenz in der Automatisierung. ELEKTRONIK 16 (1984) 8, S. 115
/2/ Lehmann, R. u. a.: Alle Achsen im Griff. ELEKTRONIK 18 (1985) 9, S. 126
/3/ Langmann, R., Mergner, M.: Zählschaltung für rechnergesteuerte Positionierantriebe. Radio, Fernsehen, Elektronik, Berlin 35 (1986) 2, S. 93
/4/ Langmann, R.: Einsatz von Schrittmotoren in momentgestellten Antrieben. Feingerätetechnik, Berlin 35 (1986) 5, S. 205
/5/ Langmann, R.: Mikroprozessorgesteuertes Positioniersystem. Wirtschaftspatent, WP G05B/ 300091.8

☒ KONTAKT ☏

VEB Robotron-Rationalisierung Weimar, Objekt Erfurt, Abt. EE2, Kaufmännerstr. 8b, Erfurt, 5020

BC A 5110 mit CP/M-kompatiblem Betriebssystem

Die Popularität CP/M-kompatibler Betriebssysteme hat zu einem schnellen moralischen Verschleiß der PROM-Version des BC A 5110 beigetragen, die in ihrer Grundvariante nur für das Betriebssystem BROS ausgelegt war. Die vom VEB Kombinat Robotron empfohlene Umrüstung des A 5110 auf eine RAM-Version ist relativ aufwendig und kommt deshalb für viele Kunden nicht in Frage. Das an der Ingenieurhochschule Dresden (jetzt Informatikzentrum an der Technischen Universität) entwickelte Betriebssystem IHD/CP ist auch für den A 5110 ohne große Hardwareänderungen einsetzbar. Das Betriebssystem BROS wird auch weiterhin durch eine Kaltstart-Diskette zur Verfügung stehen. In Verbindung mit dem CAS-PASCAL-Compiler wird den Nutzern von BROS damit ein relativ problemloser Übergang auf ein modernes Betriebssystem ermöglicht.

Komponenten von IHD/CP

IHD/CP ist für K-1520-OEM-Rechner und den BC A 5110 ausgelegt. Es ist in seinen Datenformaten CP/A- (und damit auch SCP-) kompatibel und hat folgende Vorteile:
- automatische Formaterkennung und -bearbeitung der Diskettenformate 26 Sektoren zu 128 Byte pro Spur, 16 Sektoren zu 256 Byte pro Spur, 8 Sektoren zu 512 Byte pro Spur und 4 Sektoren zu 1024 Byte pro Spur
- Speicherkapazität pro 8″-Diskette von 296 KByte für Systemdisketten, 308 KByte für Standarddisketten und 346 KByte für Spezialanwendungen durch ein Sonderformat (9 × 512)
- modularer Aufbau, dadurch leichte Erweiterbarkeit um weitere E/A-Geräte
- „frei" ladbares System, das heißt, auch andere Betriebssysteme können geladen werden (z. B. UDOS).

Folgende Programme gehören zur Umrüstung:
- @ IHD_CP.SYS Betriebssystem
- SYSLAD.COM Systemlader
- BOOTLAD.COM Bootlader
- BOOTGEN.COM Generierungsprogramm für Systemlader
- BROS.LAD Kaltstart des Betriebssystems BROS.

Erforderliche Änderungen der Hardware des BC A 5110
- ZRE K 2525
- Entfernen der PROMs und Bestückung mit 2 PROM U 555 (Anfangslader)
- Wickelbrücken ändern
- OPS
- (BLP 05–320–0121–4); es sind volle 64 KByte erforderlich, das heißt, unter Umständen muß eine Nachrüstung der freigebliebenen Plätze mit U 256 erfolgen bzw. es ist eine 16 KByte-RAM-BLP zusätzlich auf den freien Steckplatz einzufügen.
- Wickelbrücken ändern. Die PROM-Kassette wird nicht mehr benötigt und kann für andere Zwecke genutzt werden.

Einsatz von IHD/CP für K–1520–OEM-Rechner

Für OEM-Rechner kann IHD/CP für folgende Konfigurationen generiert werden:
- ZRE K 2521, K 2525
- ATS K 7028 oder ATD K 7026 zum Anschluß von Tastatur und Drucker
- AFS K 5121 für 8″-Laufwerke
- ABS K 7023 oder ABS K 7024 für BAB 1 oder BAB 2.

Außerdem können weitere K-1520-Steckeinheiten in das System eingefügt werden, z. B.:
ADA K 6022 für LB-Leser, LB-Stanzer, Drucker mit Centronics-Interface
AKB K 5020 für Anschluß eines Magnetbandkassettengerätes.
PPE K 0420 für PROM-Programmiereinheit
VIS2A für Grafikprogramme

Gert Aurig/Manfred Roth

☒ KONTAKT ☏

Informatik-Zentrum des Hochschulwesens an der TU Dresden, Informatik-Rechenzentrum, Mommsenstr. 13, Dresden, 8027; Tel. 4575503.

REDABAS

Arbeit mit Datenbanken (Teil II)

Dr. Thilo Weller, Matthias Donner
Karl-Marx-Universität Leipzig,
Sektion Wirtschaftswissenschaften/
Organisations- und Rechenzentrum

- Diskettenzugriffe, insbesondere beim indizierten Zugriff: Wenn die Datenbank oder/und die Index-Dateien durch häufige Datenergänzung und -modifikation auf dem externen Speicher nicht mehr in hintereinanderliegenden Blöcken verteilt ist bzw. sind, verzögern sich Diskettenzugriffe.
Dies läßt sich dadurch beseitigen, daß die Diskette unter Betriebssystemkontrolle mittels der PIP-Kommandos unter SCP kopiert wird, wobei getrennt abgelegte, zu einer Datei gehörende Blöcke wieder direkt hintereinander gespeichert werden und die Zugriffszeiten bei REDABAS sich verkürzen.
- Die Zugriffszeit beim Arbeiten mit indexsequentiellen Zugriffen wird weiter optimiert, wenn die Datenbank selbst in der Reihenfolge vorsortiert ist, wie es sonst durch den Indexhauptschlüssel geschieht. Vor allem bei Daten geringer Fluktuation (Stamm-Daten) sollte dies beachtet werden. Die Sortierung läßt sich leicht erreichen, indem die Datenbank mit der gewünschten Index-Datei aktiviert und dann in eine neue Datei kopiert wird.
Anschließend kann z. B. die bisherige Datenbank gelöscht, die neue umbenannt und mit dieser auch ohne Index-Datei weitergearbeitet werden.
- Die Nutzung der COPY-Anweisung unter REDABAS sollte nur bei entsprechend verwendeten Optionen wir Feldauswahl u. ä. eingesetzt werden; sie ist wesentlich zeitaufwendiger als z. B. PIP unter SCP.

10. Schnittstelle von REDABAS zu weiterer Software

Für die Kommunikation zwischen REDABAS und anderer Standardsoftware (z. B. Textverarbeitungsprogramm TP, Kalkulationsprogramm KP) bzw. Anwenderprogrammen, z. B. BASIC oder PASCAL, steht eine Schnittstelle auf Datenebene zur Verfügung:

Standardsoftware bzw. Anwenderprogramme $\overset{\text{SDF}}{\longleftrightarrow}$ REDABAS

REDABAS ist mit diesem SDF (System-Daten-Format; ASCII-Standard-Datenformat) in der Lage,
– Datenbankkopien in einem Format zu erzeugen, die auch von anderen Anwenderprogrammen oder anderer Standardsoftware verarbeitet werden können
– Dateien selbst zu verarbeiten, die durch solche Programme erstellt werden.
Der Typ dieser Dateien ist eine Textdatei, deren Dateninhalt aus beliebigen Zeichen des ASCII-Codes besteht und jeder Datensatz mit Wagenrücklauf und Zeilenvorschub (CR/LF) abgeschlossen ist. Die Datensätze sind sequentiell angeordnet und können konstant oder variabel sein.

10.1. Bereitstellung von REDABAS-Datenbanken

COPY TO ‹dateiname› [SDF]
[DELIMITED] [WITH ‹trennzeichen›]

REDABAS erzeugt damit eine Kopie der aktivierten Datenbank vom Typ Textdatei (dateiname. TXT) im ASCII-Standard-Datenformat. Das Ausgabeformat der Sätze hängt davon ab, welche Optionen in der COPY-Anweisung verwendet werden:

– COPY TO ‹dateiname› SDF

Die Felder werden in voller, definierter Länge *ohne* Begrenzungs- und Trennzeichen in der Textdatei angeordnet. Numerische Werte werden innerhalb der Felder rechtsbündig, Zeichenketten linksbündig dargestellt.

– COPY TO ‹dateiname› DELIMITED

Alle Felder im Ausgabesatz sind durch Kommas voneinander getrennt. Zeichenketten sind in Hochkommas eingeschlossen, endständige Leerstellen werden entfernt. DELIMITED schließt SDF automatisch ein.

– COPY TO ‹dateiname› DELIMITED WITH ‹trennzeichen›

Führende Leerstellen in numerischen Feldern und endständige Leerstellen bei Zeichenkettenfeldern werden entfernt. Die Felder sind durch das in der WITH-Option angegebene (d. h. dem WITH unmittelbar folgende) Trennzeichen (Sonderzeichen) voneinander getrennt und enthalten keine zusätzlichen Begrenzungszeichen.

Soll von der Ausgangsdatenbank nur ein Teil (z. B. ausgewählte Feldnamen, Datensätze) in die Textdatei übernommen werden, ist die COPY-Anweisung mit den entsprechenden Optionen zu nutzen:

```
COPY TO ‹dateiname› [‹bereich›]
          [FIELD ‹feldnamen›]
      [FOR ‹bedingung›] [SDF]
          [DELIMITED
          [WITH ‹trennzeichen›]]
```

10.2. Übernahme von Dateien in REDABAS

APPEND FROM ‹dateiname› [FOR ‹bedingung›] [SDF] [DELIMITED]

REDABAS verarbeitet Textdateien, die von anderen Programmen erzeugt wurden, mit der erweiterten APPEND-Anweisung. Dabei werden die Datensätze aus der Datei dateiname (im ASCII-Standard-Datenformat) in die aktivierte Datenbank übernommen. Die Reihenfolge der Felder in der Textdatei muß der definierten Feldanordnung in der aktiven Datenbankdatei entsprechen, an die die Datensätze der Textdatei angefügt werden.
Analog zur COPY-Anweisung können die Datenfelder unterschiedlich begrenzt sein:
– APPEND FROM ‹dateiname›
[FOR ‹bedingung›] SDF

Die Felder in der Textdatei werden in der gleichen Länge erwartet, wie sie in der aktiven Datenbankdatei definiert sind. Die Datenfelder müssen in der Textdatei ohne Trenn- und Begrenzungszeichen angeordnet sein.

– APPEND FROM ‹dateiname›
[FOR ‹bedingung›] DELIMITED

REDABAS erwartet die Datenfelder in der Textdatei durch Kommas voneinander getrennt. Felder vom Typ Zeichenkette können in Hochkommas oder Anführungszeichen eingeschlossen sein. Andere Begrenzungszeichen werden zum Feldinhalt gehörig betrachtet.

10.3. Beispiel Schnittstelle REDABAS – TP

Mittels des Textprogramms TP soll ein Serienbrief erstellt werden, der für jeden Empfänger passend gestaltet ist.
Die entsprechenden Angaben des Empfängers sollen einer REDABAS-Datei ADRESSEN.DBD folgender Struktur entnommen werden.

FELD	NAME	TYP	LÄNGE	DEZ
001	NAME	C	010	
002	VORNAME	C	008	
003	STRASSE	C	015	
004	ORT	C	008	
005	PLZ	N	004	

1. Schritt

Konvertierung (Kopieren) der REDABAS-Datei ADRESSEN.DBD in eine Textdatei ADRESSEN.TXT mit Kommas als Trennzeichen (TP verarbeitet als Trennzeichen zwischen Datenfeldern Kommas).
USE ADRESSEN
COPY TO ADRESSEN DELIMITED WITH,

2. Schritt

Erstellen einer Textschablone mit TP
VEB Gebäudewirtschaft Dresden
· op (Ausschalten der Seitenwechselanzeige)
· df B: ADRESSEN.TXT
(Angabe der entsprechenden Datei, hier im Laufwerk B)
· rv NAME, VORNAME,
STRASSE, ORT, PLZ
(Angabe der Feldnamen; Übereinstimmung mit der Ausgangsdatenbank bzgl. Reihenfolge beachten!)
Familie
& NAME & & VORNAME & (Markierung

& STRASSE & der Textstellen, die später
& ORT & durch konkrete
& PLZ & Werte ersetzt werden)

Dresden, 30. März 1987

Werte Familie & Name &!
Wir möchten Sie bitten, zur Überreichung des Mietvertrages . . .

. . .

. . .

3. Schritt

Ausdruck der Serienbriefe mit TP (Option M: Kombo-Druck)

Beim Ausdruck wird jeder Variablenname durch einen Variablenwert (aus der Datei ADRESSEN.TXT) ersetzt:

VEB Gebäudewirtschaft Dresden
Familie
Lehmann Kurt
Hauptstr. 15
Dresden
8060

Dresden, 30. März 1987

Werte Familie Lehmann!
. . .

10.4. Schnittstelle REDABAS – Höhere Programmiersprache

Für diese Schnittstelle wird als Beispiel die Programmiersprache PASCAL verwendet (System PASCAL 880/S /5/, Sprache kompatibel zu TURBO-PASCAL).
Die Schnittstelle kann nützlich sein, um bestimmte Teilaufgaben (z.B. Sortierung) in dieser höheren Programmiersprache und damit wesentlich schneller zu realisieren. Als Ausgangspunkt dient wiederum die Datenbank ADRESSEN.DBD mit analoger Struktur (siehe 10.3.). Ziel soll es sein, die Datensätze dieser Datei unter Pascal verfügbar zu machen.

1. Schritt

Konvertieren (Kopieren von ADRESSEN.DBD in eine Textdatei ADR.TXT im SDF-Format.
USE ADRESSEN COPY TO ADR SDF

2. Schritt

Erstellen eines PASCAL-Programms zur Übernahme dieser Datensätze aus der temporären Datei ADR.TXT in die Datei ADR.DAT

```
program REDABAS_TURBO_DATEI_CONVERTIERUNG
type                    {Vereinbarung spezi-
                         fischer Datentypen}
  Adresse = record
        Name    : string [10];
        Vorname : string [8];
        Straße  : string [15];
        Ort     : string [8];
    end;
AdressDatei = File of Adresse;
var
  Datensatz: Adresse
             {Vereinbarung von Programm-
              variablen}
  Datei    : AdressDatei;
  TF       : text;
```

Thilo Weller ist Hochschuldozent an der Sektion Wirtschaftswissenschaften der Karl-Marx-Universität Leipzig. Nach dem Studium der Physik an der KMU 1967–1974 arbeitete er bis zur Promotion B auf dem Gebiet rechenintensiver molekülphysikalischer Untersuchungen. Mit der breiteren Entwicklung der Mikrorechentechnik wandte er sich dann Vorlaufuntersuchungen zu dieser Problematik in gesellschaftswissenschaftlichen Fragestellungen zu und befaßt sich derzeit im Rahmen der Wirtschaftsinformatik mit Fragen des Hard- und Softwareeinsatzes vernetzter Arbeitsplatzrechner.

Matthias Donner ist wissenschaftlicher Assistent am Organisations- und Rechenzentrum der Karl-Marx-Universität Leipzig. Er studierte von 1977–1981 an der Technischen Hochschule Leipzig Technische Kybernetik und Automatisierungstechnik.
In seiner Tätigkeit am ORZ der KMU widmet er sich insbesondere dem Problemkreis der Mikrorechentechnik, speziell deren Nutzung zur digitalen Simulation.

```
begin
  ClrScr;
  assign(TF, 'ADR.TXT'); reset(TF);
                  {Aktivieren von ADR.TXT zum
                   Lesen}
  assign(Datei, 'ADR.DAT'); rewrite(Datei);
                  {Generierung von ADR.DAT}
  while not EOF(TF) do
    begin
      with Datensatz do
        begin
          read(TF, Name);
                     {Lesen der einzelnen
                      Datensatz-Felder}
          read(TF, Vorname);
          read(TF, Straße);
          read(TF, Ort);
          read(TF, PLZ)
                 {Letztes Feld mit readln
                  lesen}
        end;
      write(Datei, Datensatz);
                 {Schreiben des Datensat-
                  zes in ADR.DAT}
      write('–',Datensatz,Name)
                 {Protokoll}
    end;
  close(Datei); {Schließen der Adreßdatei
                 ADR.DAT}
  close(TF);    {Schließen der Textdatei
                 ADR.TXT}
end.
```

3. Schritt

Weiterbearbeitung der REDABAS-Adreßdatei unter PASCAL 880/S.

10.5. Schnittstelle höhere Programmiersprache – REDABAS

Als Beispiel soll eine unter PASCAL 880/S bearbeitete Adreßdatei ADR.1DAT für REDABAS verfügbar gemacht werden.

1. Schritt

Entwickeln eines PASCAL-Programms zum Kopieren der Datensätze von ADR.1.DAT in eine temporäre Textdatei ADR1.TXT (Die Programmversion soll *eine* Möglichkeit aufzeigen).

```
program TURBO_REDBAS_DATEI_CONVERTIERUNG;
type                    {Vereinbarung spezi-
                         fischer Datentypen}
  Adresse = record {Datensatz-Vereinba-
                    rung}
    Name    : string [10];
    Vorname : string [8];
    Straße  : string [15];
    Ort     : string [8];
    PLZ     : string [4];
    end;
  Adress-Datei = File of Adresse;
var                {Vereinbarung von Pro-
                    grammvariablen}
  Datensatz: Adresse;
  Datei    : AdressDatei;
  TF       : text;
  i        : integer;

begin
  ClrScr;
  assign(Datei, 'ADR1.DAT'); reset(Datei);
                  {Aktivieren von
                   ADR1.DAT zum Lesen}
  assign(TF, 'ADR1.TXT'); rewrite(TF);
                  {Generierung von
                   ADR1.TXT}
  while not EOF(Datei) do
    begin
      read(Datei, Datensatz);
                  {Lesen eines Daten-
                   satzes aus ADR1.DAT}
      with Datensatz do
      begin       {Schreiben der Daten-
                   satzfelder}
      write (TF,Name);
      if length(Name) < 10 then
        for i: = length(Name)+1 to 10
             do write(TF,' '); {Auffüllen
                               mit Leerzeichen}
      write(TF, Vorname);
      if length(Vorname) < 8 then
        for i: = length(Vorname)+1 to 8
             do write(TF, ' ');
      write(TF, Straße);
      if length(Straße) < 15 then
        for i: = length(Straße)+1 to 15
             do write(TF,' ');
      write(TF, Ort)
      if length(Ort) < 8 then
        for i: = length(Ort)+1 to 8
        do write(TF, ' ');
      writeln(TF,PLZ);
    end;
    write('–',Datensatz.Name);
                  {Protokoll}
  end;
  close(Datei);  {Schließen der Adreßdatei
                  ADR1.DAT}
  close(TF);     {Schließen der Textdatei
                  ADR1.TXT}
end.
```

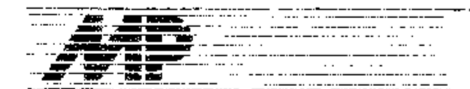
2. Schritt

Kopieren der Struktur von ADRESSEN.DBD nach ADR1.DBD (in REDABAS)
USE ADRESSEN
COPY STRUCTURE TO ADR1

3. Schritt

Anfügen der Datensätze aus der temporären Textdatei ADR1.TXT an ADR1.DBD
USE ADR1
APPEND FROM ADR1.TXT SDF

Fortsetzung auf Seite 19



Kurs

Fortsetzung von Seite 14

11. Der interne Aufbau einer Datenbank

Für die Untersuchung des internen Aufbaus einer beliebigen Datei – also auch einer Datenbank – können Debugger wie DU oder andere Monitorprogramme genutzt werden. Eine solche Betrachtung gibt Auskunft über die REDABAS-interne Verwaltung der Strukturvereinbarungen (Tafel 1).

Tafel 1

←	REDABAS – Datenbank		→
Struktur-vereinbarung	1. Datensatz	...	n. Datensatz

Byte 0 – 519

REDABAS reserviert die ersten 520 Bytes für die Beschreibung der Datenbank bzw. der Datenfelder. Im ersten Byte (Byte 0) steht mit 02h ein Wert, der wohl vordergründig aus Kompatibilitätsgründen seine Daseinsberechtigung hat. Die nächsten zwei Bytes enthalten die Gesamtzahl der Datensätze, wozu anzumerken ist, daß das niederwertige *vor* dem höherwertigen Byte gespeichert ist. In den nächsten drei Bytes steht das Datum der letzten Änderung der Datenbank in der Form TT/MM/JJ (hexadezimal). Die um eins erhöhte Länge eines Datensatzes ist in den Bytes 6 und 7 gespeichert. Soweit die allgemeinen Angaben.
Die restlichen Bytes (Byte 8 bis 519) sind für die Beschreibung der n Datenfelder (1 < = n < = 32) vorgesehen (Tafel 2).

Tafel 2

Byte	Wert	Bedeutung
00	02h	
01–02	xx xx	Anzahl der Datensätze
03–05	xx xx xx	Datum (TT MM JJ)
06–07	xx xx	Datensatzlänge + 1
08–519		Strukturdefinitionen
520–n	20h	Beginn Datenbereich (1. Datensatz)
n + 1	1Ah	Dateiende

Zur Feldbeschreibung dient jeweils ein 16-Byte-Bereich, der alle zur eindeutigen Festlegung der Struktur erforderlichen Definitionen enthält (Tafel 3).

Tafel 3

Byte	Wert	Bedeutung
00–09		Feldname
10	00h	
11	xx	Feldtyp
12	xx	Feldlänge
13–14		reserviert
15	xx	Anzahl Dezimalstellen

Ein 0Dh im Byte 0 des 16-Byte-Bereichs kennzeichnet den Abschluß der Strukturdefinitionen.
Der Datenbereich beginnt mit dem Byte 520. Am Anfang eines jeden Datensatzes steht ein Leerzeichen (20h), dem sich die Daten entsprechend der Strukturvereinbarung anschließen. Abgeschlossen wird die Datenbank mit dem Dateiendekennzeichen 1Ah.
Das nachfolgende PASCAL-Programm analysiert die Struktur einer beliebigen REDABAS-Datenbank und bringt diese auf dem Bildschirm zur Anzeige.
Dieses einfache PASCAL-Demonstrations-Programm zeigt die Möglichkeit der Weiterverarbeitung von REDABAS-Datenbank-Informationen unter Nutzung einer höheren Programmiersprache auf.

```
Program Redabas_Datenbank_Struktur;
const
  alphanum : set of char =
                  Ä'A'..'Z','_','0'..'9'ü;
type
  eintrag = record
              name: stringÄ16ü;
              laenge: byte;
              typ: (C, N, L)
            end;
  zeile = stringÄ80ü;
  bytearray = arrayÄ0..520ü of byte;
var
  quelle,ziel : file;
  puffer : bytearray;
  dateistruktur : record
              eintrzahl:integer;
              eintr:arrayÄO..127ü
                      of eintrag
            end;
  quellname,zielname : stringÄ30ü;
  gesamtzahl,satzlänge, anzahlptr,
  anfangptr,offset,längenoffset,
  typoffset : integer;
  Redabas_Byte_0 : byte;
procedure fehlernachricht (meldung:zeile);
begin writeln(meldung); halt end;
procedure analyse;
var
  i, j:integer;
  lowbyte,highbyte,typflag,letter:byte;
  begin (* analyse *)
    blockread(quelle,puffer,4);
(* liest die ersten 512 Byte der Datei *)
    Redabas_Byte_0:=pufferÄ0ü
    if Redabas_Byte_0=2 then
  begin
    anzahlptr:=0;anfangptr:=8;
    offset:=16;typoffset:=11;
    längenoffset:=12
      end
    else fehlernachricht
              ('Keine Redabas-Datei');
    lowbyte:=pufferÄ1ü;
    highbyte:=pufferÄ2ü;
    gesamtzahl:=(highbyte shl 8)
                  or lowbyte;
    lowbyte:=pufferÄ6ü;
    highbyte:=pufferÄ7ü;
    satzlänge:=(highbyte shl 8)
                or lowbyte;
    dateistruktur.eintrzahl:=0;
    writeln(Gesamtzahl, 'Einträge a',
          satzlänge, 'Byte.');
    writeln;
    writeln('Feld                Länge Typ');
    writeln;
    with dateistruktur do
      repeat
        i:=(eintrzahl*offset)+anfangptr;
        j:=i;
        letter:=pufferÄiü;
        if letter <>$0D then
        with eintrÄsucc (eintrzahl)ü do
          begin
            name:=chr (letter);
            repeat
              j:=succ (j);
              letter:=pufferÄjü;
              if chr (letter)
               in alphanum then
               name:=name+chr(letter)
            until not
              (chr(letter) in alphanum);
            typflag:=puffer
                    Äi+typoffsetü;
            länge:=puffer
                    Äi+längenoffsetü;
            write(name,
              '':12-length(name),
              länge:4,
              chr(typflag):4);
            case chr(typflag) of
              'C' : typ:=C;
              'N' : typ:=N;
              'L' : typ:=L;
              else typ:=C
            end;
            eintrzahl:=succ(eintrzahl);
            writeln
          end
      until letter=$0D;
    writeln
  end;    (*  analyse  *)

BEGIN
  writeln
  ('REDABAS – Datenbank-Informatio-
  nen');
  writeln; write('Datei :');
  readln(quellname);
  if pos('.',quellname)=0 then
    quellname:=quellname+'.DBD';
  assign(quelle, quellname);
  (*I–*)
  (* schaltet Fehlerbehandlung *)
  (* durch das System aus *)
  reset(quelle);
  if ioresult<>0 then
    fehlernachricht
    ('Keine gültige Datei?');
  analyse;
  close(quelle)
END.
```

Schluß

Literatur

/1/ REDABAS Programmbeschreibung, Systemunterlagendokumentation. VEB Robotron-Projekt, Dresden, 1985

/2/ Eggerichs, W.: dBASE II (Bd. 1 und 2). VEB Verlag Technik, Berlin, 1986

/3/ Eggerichs, W.: dBASE II. Band 3: Aufbau und Nutzung von Datenbanken. Dr. A. Hüthig Verlag, Heidelberg 1986

/4/ Donner, M; Weller, Th.: Hochschul-Folien/-Dia-Reihe Datenbanksystem REDABAS. HFR 837/858/872, HR 1494/1515/1529. Institut für Film, Bild und Ton, Berlin, 1987

/5/ Bedienungsanleitung und Sprachbeschreibung für das Programmiersystem PASCAL 880/S unter Steuerung des Betriebssystems SCPX, Systemunterlagendokumentation. VEB Robotron Büromaschinenwerk, Sömmerda, 1987

Erkennung von Eingabefehlern in REDABAS-Programmen

Oskar Schönherr, Karl-Marx-Stadt

Beim Einsatz von Programmen ist es insbesondere notwendig, eine störungsfreie Arbeit auch dem Nutzer zu garantieren, der das Programm selbst nicht erstellt hat und dessen inneren Aufbau nicht kennt. Es muß erreicht werden, daß möglichst viele Fehleingaben vom Programm selbst erkannt werden. Bei der Arbeit mit REDABAS wurden dazu vom Autor eine Reihe besonders für Einsteiger geeignete Programmroutinen verwendet, die auf Grund ihrer universellen Anwendbarkeit und ihres einfachen Aufbaus für viele Anwendungsfälle leicht nachzuvollziehen sind. Es wurde davon ausgegangen, daß die Fehlererkennung bereits bei der Eingabe erfolgen sollte und nicht erst in der weiteren Programmabarbeitung, um Rücksprünge in vorangegangene Programmteile zu vermeiden. Günstig für die Bearbeitung war die Tatsache, daß sich mit REDABAS alle Programme in Menütechnik aufbauen lassen. Im folgenden werden die in den Bildern 1 bis 3 dargestellten Programmteile kurz in ihrer Wirkung beschrieben. Die einzugebende Zeichenvariable in den Beispielen heißt zvar.

1. Auswahl aus einem Menü

Wenn aus einem Menü eine Funktion ausgewählt werden soll, garantiert die im Bild 1 vorgestellte Form, daß nur die Möglichkeiten, die das Menü vorsieht, auch als echte Eingaben angenommen werden. Alle anderen Eingaben erscheinen auf dem Bildschirm, führen aber nicht zum Verlassen der Schleife. Bei einer Erweiterung des Menüs kann durch eine Veränderung der Zeichenkette die DO-WHILE-Schleife schnell auf die neuen Bedingungen angepaßt werden.

```
C:BEISP1  .PRG

*
* Auswahl aus einem Menue
*
ERASE
@  6,20 SAY '1 - Eingabe'
@  7,20 SAY '2 - Aenderung'
@  8,20 SAY '3 - Recherche'
@  9,20 SAY '4 - Austragen'
@ 10,20 SAY 'E - Ende'
@ 12,30 SAY 'Auswahl'
STORE ' ' TO zvar
DO WHILE .NOT. zvar$'1234E'
  @ 12,40 GET zvar
  READ
ENDDO
*
* Weiter mit Fallunterscheidungen
*
```

Bild 1 Funktionsauswahl

2. Auswahl längerer Zeichenketten durch ein Hilfsmenü

Oft besteht die Notwendigkeit, längere Zeichenketten einzugeben, die immer wieder auftauchen. Dabei ist eine garantiert gleiche Schreibweise gefordert, wenn nach diesen Zeichenketten in Dateien recherchiert werden soll. Bewährt hat sich dabei die in Bild 2 vorgestellte Variante, die in einer angenommenen Datei mit Berufsgruppen die Recherche mit immer gleichen Zeichenketten (Frage der Schreibweise bei verschiedenen Anwendern bzw. Verhinderung von Eingabefehlern) gewährleistet.
Zur Bedienungsfreundlichkeit wurde die am meisten verwendete Berufsgruppe vorgewählt und kann mit ET angefordert werden. Zusätzlich wird deutlich, daß es damit auch einem Unkundigen möglich ist, das Programm abzuarbeiten, ohne Fehler zu provozieren. Als Hilfe wird die ausgewählte Eingabe angezeigt.

```
CO=

C:BEISP2  .PRG

*
* Auswahl laengerer Zeichenketten
*
SET TALK OFF
ERASE
@  6,20 SAY 'Auswahl der Berufsgruppe '
@  8,30 SAY '1 - Lehrling'
@  9,30 SAY '2 - Arbeiter'
@ 10,30 SAY '3 - Angestellter'
STORE ' ' TO zvar
DO WHILE .NOT. zvar$'123'
  STORE '2' TO zvar
  @ 6,45 GET zvar
  READ
ENDDO
DO CASE
  CASE zvar='1'
    STORE 'Lehrling' TO zvar
    @ 6,46 SAY zvar+CHR(20)
  CASE zvar='2'
    STORE 'Arbeiter' TO zvar
    @ 6,46 SAY zvar+CHR(20)
  CASE zvar='3'
    STORE 'Angestellter' TO zvar
    @ 6,46 SAY zvar+CHR(20) ,
ENDCASE
*
* Weiter im Programm
*
```

Bild 2 Hilfsmenü für längere Zeichenketten

3. Auswahl aus größeren Zeichenkettenmengen

Es ist in vielen Fällen nötig, daß Zeichenketten unterschiedlichster Länge aus einem Datenbestand (aus einem Feld) herausgesucht und für die weitere Bearbeitung diese zutreffenden Datensätze zur Verfügung gestellt werden müssen. Ein Beispiel dieser Art ist die Suche nach einem Namen in einer Adreßdatei. Bei Indizierung dieser Datei nach Familienname ('name') und Vorname ('vorname') muß zum Finden der Person der Name vollständig und in der Schreibweise richtig eingegeben werden.
Das im Bild 3 dargestellte Prinzipbeispiel findet in begrenzt großen Dateien auf schnelle Art den richtigen Namen, ohne daß dieser komplett eingegeben werden muß. Ähnlich funktioniert zum Beispiel die Eingabe von Reisezielen in den Schaltergeräten der Deutschen Reichsbahn.
Das vorgestellte Prinzip hat seine Grenzen in der mit Datensatzanzahl steigenden Laufzeit und bei sehr vielen gleichen Namen. Für viele Probleme ist es aber sofort anwendbar.

```
CO=

C:BEISP3  .PRG

*
* Auswahl aus groeszeren Zeichenkettenmengen (Dateien)
*
ERAS
USE c:namen INDE c:namen
SET EXAC OFF
SET TALK OFF
SET COLO OFF
SET CONF OFF
STOR ' ' TO name1,name2
STOR 30 TO A
STOR 'X' TO zvar
@ 6,10 SAY 'Versuch Schnelleingabeverfahren'
@ 7,15 SAY 'Buchstaben- oder <ET>-Taste  => weiter mit Namen
@ 8,15 SAY 'Leertaste                    => Name richtig'
@ 10,10 SAY 'Namenseingabe :'
@ 10,30 GET name2 PICT '!'
READ
CLEA GETS
FIND &name2
IF #=0
STOR ' ' TO zvar
@ 10,30 SAY 'Gesuchter Name fehlt'+CHR(20)
ELSE
@ 10,A SAY name
@ 11,A SAY vorname
STOR A+1 TO A
DO WHIL zvar#' '
STOR $(name,A-29,1) TO name1
@ 10,A GET name1 PICT '!'
READ
CLEA GETS
IF name1<>' '
SKIP
IF name2#$(name,1,A-30)
STOR ' ' TO zvar
@ 10,30 SAY 'Gesuchter Name fehlt'+CHR(20)
LOOP
ELSE
STOR name2+name1 TO name2
STOR A+1 TO A
DO WHIL name2#$(name,1,A-30).AND..NOT.EOF
SKIP
ENDD
@ 10,30 SAY name
@ 11,30 SAY vorname
LOOP
ENDI
ELSE
STOR name TO name2
@ 10,30 SAY name
@ 11,30 SAY vorname
STOR ' ' TO zvar
ENDI
ENDD zvar#'
ENDI #=0
*
* Weiter im Programm
*
* Verwendete Datei   namen.dbd
*     Feld 1     name       C  020
*     Feld 2     vorname    C  020
*     Feld 3     strasze    C  040
*     Feld 4     ort        C  040
*     usw.
*
* INDEX ON name+vorname TO namen
```

Bild 3 Auswahl aus größeren Zeichenkettenmengen

Softwareentwicklung für speicherprogrammierbare Steuerungen

Dr. Peter Löber, Dr. Günter Jaehnert, Dr. Klaus Engelmann, Harry Kreller
Technische Universität Karl-Marx-Stadt, Sektion Automatisierungstechnik

1. Einleitung

Die Grundlage jeder Automatisierung ist eine entsprechende Steuereinrichtung. Speicherprogrammierbare Steuerungen (SPS) haben sich in den letzten Jahren ein immer größeres Anwendungsgebiet erschlossen. Die obere Einsatzgrenze ist noch nicht absehbar. Weltweit werden bereits von mehr als 150 Firmen SPS-Systeme der unterschiedlichsten Leistungsklassen angeboten /1/. Das Einsatzgebiet der SPS-Systeme reicht mittlerweile von der Realisierung einfachster Ablaufsteuerungen bis hin zur hochgradigen Automatisierung flexibler Fertigungssysteme (FFS) durch hierarchische Steuerungsverbundsysteme /2/ und /3/. Trotz oder gerade wegen des immer größer werdenden Anwendungsgebietes existiert nach wie vor keine einheitliche bzw. standardisierte Begriffsbestimmung für SPS-Systeme /4/. Für die im weiteren betrachtete Klasse der mittleren SPS-Systeme sollen folgende Hauptmerkmale gelten:

- Bit- oder/und wortorientierte Abarbeitung von Steuerprogrammen durch einen oder mehrere Mikroprozessoren gleicher oder unterschiedlicher Verarbeitungsbreite
- Meß- und Stellsignalanzahl im Bereich von 100...1000 (einschließlich begrenzter Analogsignalerfassung und -ausgabe)
- auf die SPS begrenzter lokaler Systembus und Prozeßein- und -ausgabebus
- eigenständige Funktion der SPS mit Integrationsfähigkeit in ein Steuerungsverbundsystem
- Einbeziehung funktionserfüllender steckbarer Kartenbaugruppen zur Prozeß-, Bedien-, Leit- und Programmierebene.

Charakteristisch für SPS-Systeme ist die Tendenz der stagnierenden bzw. sinkenden Hardwarekosten bei überdurchschnittlich steigenden Softwarekosten sowie die ständige Erhöhung der Leistungsfähigkeit von SPS. Die steigenden Softwarekosten sind bedingt durch die wachsende Komplexität der zu automatisierenden Prozesse, die unzureichende Effizienz bei der Herstellung von Softwarekomponenten und die mangelnde Portabilität von Softwareprodukten bei einer für den Anwender schwer überblickbaren SPS-Vielfalt. Im vorliegenden Beitrag wird gezeigt, welche Probleme bei der Softwareentwicklung für SPS auftreten und welche Mittel und Wege zur Reduzierung der Entwicklungskosten für System- und Anwendersoftwareprodukte führen können.

2. Grundstruktur von SPS-Systemen

Unter der Grundstruktur eines SPS-Systems soll prinzipiell die im Bild 1 angegebene Struktur verstanden werden. Außer der eigentlichen Steuereinrichtung, bestehend aus Prozeßein- und -ausgabebaugruppen, der Verarbeitungseinheit und dem Steuerprogramm, wird in zunehmendem Maße der Anschluß rechentechnischer Geräte ermöglicht, der einmal die SPS zu „kleinen" Prozeßrechnern qualifiziert und zum anderen rechnergestützte Programmierung und Inbetriebnahme zuläßt. Die Handhabung dieser Technik stellt derzeit beim Anwender das Hauptproblem dar. Es wird noch dadurch verschärft, daß sowohl national als auch international immer leistungsfähigere SPS in kurzen Zeiträumen angeboten werden und die bereitgestellten SPS-Betriebssysteme und Programmiermethoden immer anspruchsvoller werden /6/, /7/, /8/. Des weiteren ist zu erkennen, daß die Standardisierung dieser Entwicklung nur schwer folgen konnte /4/. Für die Programmierung mit Fachsprachen zeichnet sich eine Lösung dieser Problematik durch die IEC Sec. 65A (1985) /9/ ab.

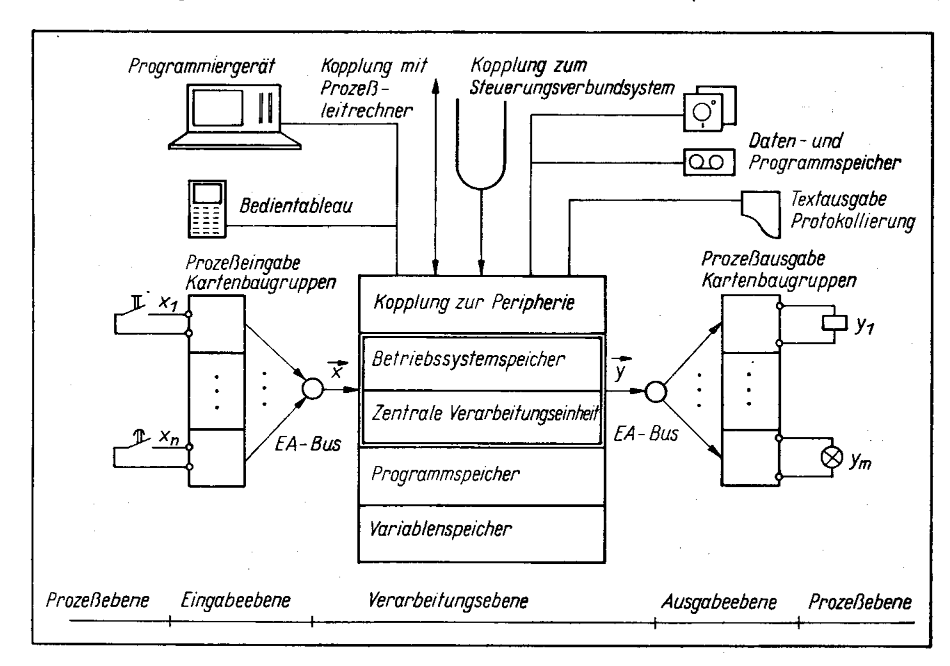


Bild 1 Allgemeine Grundstruktur eines SPS-Systems

3. Systemsoftware–Bestandteile und Aufgaben

Das Betriebssystem einer SPS auf Mikrorechnerbasis umfaßt die folgenden Komponenten:

- Initialisierung des Rechnersystems
- Diagnose von Bausteinen und Baugruppen
- Echtzeitsteuerung
- Treiber für E/A-Kartenbaugruppen und für periphere Geräte
- Inbetriebnahmeunterstützung.

Diese Komponenten adressieren während des Betriebes direkt oder indirekt über ein Datenfeld (Generierdaten), welches im Ergebnis der Projektierung vorliegt, die konkrete Hardwareumgebung. Betriebssystemkomponenten und Datenfeld sind aufeinander abgestimmt. Die Projektierungsrichtlinie des Herstellers ist deshalb genau einzuhalten, z. B. /10/ und /11/. Im einzelnen erfüllen die Betriebssystemkomponenten folgende Funktionen:

1. Initialisierung

Mittels Initialisierung aller programmierbaren Ein-/Ausgabeeinheiten (z. B. CTC, SIO, PIO, EA-Kartenbaugruppen) sowie durch Einstellen von Prozessorzustand (Register, Interruptsystem) und RAM-Speicher wird der definierte Anlauf der SPS gewährleistet.

2. Diagnose

Die Diagnose kann in 3 Phasen eingeteilt werden:

- Einschaltdiagnose

Alle vom Prozessor ansprechbaren Bauelemente bzw. Kartenbaugruppen werden auf grundsätzliche Funktionstüchtigkeit geprüft. Das Testergebnis wird nach außen angezeigt. Im Fehlerfall ist der Steuerungsbetrieb verriegelt.

- On-line-Diagnose (mit Prozeßanschluß)

Während des Betriebes werden wichtige Parameter ständig überwacht (z. B. Spannungsausfall, Not-Aus-Schleife, Zykluszeit). Fehler führen in der Regel hier zum Abschalten der Steuerspannung der Prozeßstellsignale. Daraus folgt für den Anwender, die Prozeßanpassung so vorzunehmen, daß ein Abschalten keine Havarien zur Folge hat.

- Off-line-Diagnose (ohne Prozeßanschluß)

Es werden zur Fehlerlokalisierung spezielle umfangreiche statische und dynamische Prüfungen vorgenommen (z. B. Kartenbaugruppentest, Reeingabe, elektronische Sicherung, Test auf Wärmeempfindlichkeit und elektromagnetische Verträglichkeit).

3. Echzeitsteuerung

Die Echtzeitsteuerung bildet das Kernstück des Betriebssystems. Sie organisiert das

Einschalten von Betriebssystemroutinen zwecks Ansteuerung und Datenaustausch mit der E/A-Peripherie und interner Datenmanipulation und -verwaltung zur Realisierung der Steuerungsanweisungen. Die Echtzeitsteuerung kann als ein Zweiebenensystem betrachtet werden. Die erste Ebene organisiert eine Taskverwaltung. Sie ist dann notwendig, wenn Ablaufsteuerung (Steuerung des SPS-Betriebes) sowie Kommunikations-, Regelungs-, Protokollier- und Überwachungsprozeduren weitgehend unabhängig voneinander arbeiten. Auf dieser Ebene sind spezielle Operationen zur Tasksteuerung und zur Datenübergabe erforderlich. Die Ablaufsteuerung ist ebenfalls eine Task, die auf der zweiten Ebene die typischen SPS-Operationen realisiert. Diese Ablaufsteuerung kann vollständig als Interpreter mit festem, vom Programmierer unbeeinflußbarem, Zyklus (Einlesen – Ausgeben – Verarbeiten), wie etwa beim SPS-System PC600 /10/, realisiert sein oder auch als Teilinterpreter für ausgewählte Steueranweisungen, wie EA-Anweisung, Steueranweisungen für Softwarezeitglieder, wie z. B. SPS-System MRS700 /11/. Da im letzteren Fall das Steuerprogramm im Maschinenkode vorliegt und weitgehend direkt abgearbeitet wird, verkürzt sich die Verarbeitungszeit erheblich im Vergleich zur interpretativen Abarbeitung. Entgegengesetzt dazu verhalten sich allerdings der Speicherplatzbedarf und der Übersetzungsaufwand des Steuerprogramms. Beide Verarbeitungsformen sind auf der Basis von Mikrorechnerverarbeitungseinheiten nicht steuerungsspezifisch.

4. Peripherietreiber

In SPS-Systemen sind zwei Klassen von Peripherietreibern zu unterscheiden:

- E/A-Treiber zur Ansteuerung der Prozeßein- und -ausgabekartenbaugruppen
- Treiber zur Kopplung mit Peripheriegeräten, z. B. Programmiergerät, Magnetbandkassette, Drucker.

Der E/A-Treiber ermöglicht den Datenaustausch zwischen der Verarbeitungseinheit und den E/A-Kartenbaugruppen. Da diese Baugruppen in der Regel durch eine Reihe von Besonderheiten gekennzeichnet sind, besteht der E/A-Treiber aus einem Kern, der über das Hardwaredatenfeld die E/A-spezifische Treiberroutine aufruft. Weitere Treiberbausteine werden z. B. für die Kopplung der SPS mit den Programmier- und Protokolliereinrichtungen bzw. zur Datenfernübertragung benötigt. Sie unterscheiden sich nicht prinzipiell von Treiberprogrammen für Personal- und Bürocomputer.

5. Inbetriebnahmeunterstützung

Die Inbetriebnahme von Steuerprogrammen kann durch eine Reihe von Test-Funktionen, wie statische und dynamische Anzeige des Variablenspeichers, Zwangseinstellen von Variablen, zyklusweise Programmabarbeitung sowie Echtzeitlauf mit Tracefunktion und Haltepunkten, unterstützt werden. Während der Testphase wird die Echtzeitsteuerung mit den gewünschten Funktionen beauflagt. Die Kommunikation mit dem Programmierer wird über ein an die SPS angeschlossenes Kommandogerät (Bedientableau, Personalcomputer, Programmierterminal) und die Zustandsanzeigen (LED) auf den Peripheriekarten gewährleistet.

Hardware und Betriebssystem bilden den Kern einer Basismaschine. Darauf kann mittels einer Softwareschicht programmtechnisch eine Basisschnittstelle geschaffen werden, die die Portabilität von Nutzerprogrammen (Übertragbarkeit von Programmen auf unterschiedliche Hardware) sichert, das heißt, aus der Sicht des Nutzerprogramms sind Hardware und Betriebssystem weitgehend verdeckt. Die Portabilität von Betriebssoftware ist dagegen nicht oder nur bedingt möglich. Für jede grundsätzlich neue Hardware (z. B. andere Schaltkreisfamilie) muß die Betriebssystemsoftware neu erstellt werden. Insbesondere wird die Anpassung auf andere Hardwarekonfigurationen, die die gleiche Schaltkreisfamilie nutzen, durch Bezugnahme der Betriebssystemkomponenten auf ein Hardwaredatenfeld möglich. Da die Betriebssystemsoftware der Steuerung MRS700 dieses Kriterium aufweist, konnte die Anpassung an die PC 600 vorgenommen werden.

4. Programmierung von SPS

Das Ziel bei der Programmierung besteht darin, in kurzer Zeit und mit wenig Aufwand die Steuerungsfunktion durch das Anwenderprogramm zu realisieren. Dabei bestimmen die Kriterien
- Fehlerfreiheit
- Wartungsfreundlichkeit
- Hardwarekompatibilität

maßgebend die Softwarequalität. Die Problematik der Softwareerstellung für SPS wird entscheidend durch die Einsatzspezifik geprägt:
- Wenig oder keine EDV-Kenntnisse bei den Anwendern
- Rücksichtnahme auf Prozeßanpassung
- harte Zeitforderung für den On-line-Test
- großes Spektrum der Steuerungsperipherie u. a.

Für die SPS-Programmierung sind deshalb fachbezogene Reglementierungen für Programmiertechnologie und -sprache getroffen worden, die aus Effektivitätsgründen meist ebenfalls durch spezielle Programmiergerätetechnik unterstützt werden /12/. Zur Untersetzung des Programmiervorganges sind prinzipiell fünf Arbeitsschritte zu nennen:

1. Prozeßanalyse und Spezifikation

Darunter ist die Problemerarbeitung zu verstehen. Ausgehend von der Zielfunktion des zu steuernden Prozesses und dem technologischen Schema wird die Grobstruktur der Steuerungsaufgaben entworfen. Anschließend werden im Zuge der Verfeinerung Komponenten und Funktionen definiert und die erforderlichen E/A-Schnittstellen festgelegt. Die resultierende Beschreibung der Steuerungsfunktion dient als Programmier- und Projektierungsvorlage. Nach wie vor gilt der Grundsatz, daß die exakte Prozeßanalyse unabdingbare Voraussetzung für alle weiteren Automatisierungsschritte ist.

2. Programmentwicklung

Mit Hilfe der Programmiersprache wird für die Komponenten eine funktionserfüllende Struktur, das Steuerprogramm, erstellt. Dazu werden Fachsprachen, die auf SPS-Problemklassen zugeschniten sind, verwendet. Durch sie wird der Programmentwurf wesentlich erleichtert. Anschließend erfolgt das Übersetzen und Verbinden unter Nutzung des Programmiergerätes. Danach liegt das Steuerprogramm im Maschinenkode vor. Um die Portabilität des SPS-Programms zu erreichen, wird in modularen SPS-Systemen zunächst auf ein Zwischenkodeniveau übersetzt. Dieser Zwischenkode kann in der SPS interpretiert werden (Interpreterversion) oder muß weiter bis zum Maschinenkode übersetzt werden. Er ist dann in der SPS direkt verarbeitbar (Compilierversion).

3. Programmtest

Der Programmtest hat die Aufgabe, erstens den Korrektheitsnachweis des Steuerprogramms mit der beabsichtigten Steuerfunktion zu erbringen und zweitens die Steuerfunktion hinsichtlich der Aufgabenlösung zu beurteilen. Bei Nichterfüllung einer Forderung ist der Programmiervorgang entsprechend zu wiederholen. Zum Effektivieren des Programmtests werden gezielt Hard- und Softwaremittel eingesetzt (Kopplung Programmiergerät – Steuerung, Testsoftware, E/A-Adapter, Emulator u. a.).

4. Programmdokumentation

Neben der Programmiersprache spielt die Dokumentation des Steuerprogramms eine entscheidende Rolle für Anwendung, Wartung und Programmänderungen. Prozeßbezogene Symbole, Betriebsmittelkennzeichnung durch Klartexte, Querverweise, kompakte und übersichtliche Darstellung unterstützen die Handhabbarkeit. Insbesondere muß auf die geschlossene Darstellung funktionaler Zusammenhänge geachtet werden, z. B. mit Funktionsbausteinen.

Peter Löber (42) absolvierte von 1965 bis 1969 ein Studium an der TU Dresden, Sektion Elektrotechnik. 1977 promovierte er zu Problemen der indirekten Elektronenstrahlaufzeichnungsverfahren. Danach war er als Projektant für Steuerungs- und Energieversorgungsanlagen tätig. Seit 1980 ist er Lektor an der TU Karl-Marx-Stadt, Sektion Automatisierungstechnik, und auf dem Gebiet der Modellierung und Steuerung diskontinuierlicher Systeme beschäftigt.

Günther Jaehnert (32) studierte von 1976 bis 1981 an der Sektion Automatisierungstechnik der TH Karl-Marx-Stadt, an der er anschließend ein Forschungsstudium absolvierte und 1984 auf dem Gebiet der industriellen Steuerungstechnik promovierte. Er arbeitet jetzt als wissenschaftlicher Mitarbeiter an der TU Karl-Marx-Stadt und beschäftigt sich mit netzorientierten Fachsprachen sowie deren Umsetzung auf industrielle Steuerungen.

Klaus Engelmann (30) studierte von 1975 bis 1979 an der Sektion Elektrotechnik der TU Dresden. 1983 promovierte er mit einer Arbeit zur SPS-Programmierung mit Funktionsplänen auf der Basis netztheoretisch analysierbarer Strukturen. Danach arbeitete er als wissenschaftlicher Mitarbeiter an der TU Karl-Marx-Stadt. Die derzeitigen Forschungsarbeiten behandeln Modellbildung und Simulation von Bedienungsprozessen mit Hilfe stochastischer Petrinetze.

Harry Kreller (30) studierte von 1978 bis 1983 an der Sektion Automatisierungstechnik der TH Karl-Marx-Stadt. Danach arbeitete er als wissenschaftlicher Assistent auf dem Gebiet der Steuerung von automatischen Transportsystemen in der bedienerarmen Produktion.

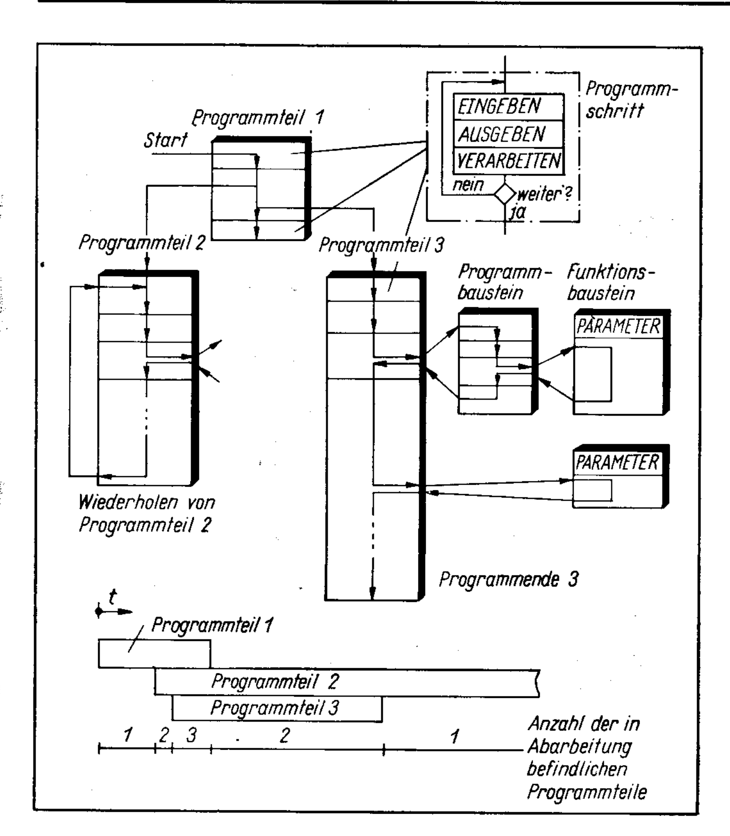


Bild 2
Beispiel für die Struktur eines MPSS-Programms und dessen Abarbeitung

Tafel 1 Übersicht zu MPSS nach /5/
var – Variablen beliebigen Formats, die mit „." gekennzeichneten Operationen sind nur mit Bitvariablen möglich
con– Konstante
zeit–Zeitangabe
sym–symbolischer Adreßbezeichner
par–Parameter, n-mal optional als Byte oder Wort oder Konstante
adr–physische Bausteinadresse
/ –lies „oder"
* –bedeutet Deaktivierung aller aktiven Prozesse, außer den veranlassenden Prozeß

VARIABLEN			KONSTANTEN	ZEITANGABEN
Typ	Format	Adresse	binär oktal dezimal hexadez ascii	Stundenbereich Minutenbereich Sekundenbereich
Eingang Ausgang Merker	Bit Byte Wort Integer	kanal.nr kanal		

OPERATIONEN				
LOGIK	VERGLEICHE	ZÄHLOP.	ZEITOP.	UNTERPROGR.
.U var .UN var .O var .ON var U(UN(O(ON( ))N .= var .=N var .R var .S var	GL var/con GLN var/con GRG var/con KLG var/con KL var/con	ZV var ZR var	.TI var zeit .TE var zeit .TA var zeit .TR var	BA sym BAB sym BABN sym .BAV var sym .BAVN var sym BE BEB BEBN .BEV var .BEVN var BAF adr par
	ARITHMETIK	SPRUNGOPERATIONEN		
	ADD var/con SUB var/con MUL var/con DIV var/con MOD var/con	SP sym SPB sym SPBN sym .SPV var sym .SPVN var sym		

LADEOP.	EIN/AUSGABE	STEUERUNG DES LOG. MEHRPROZ.SYST.
L var/con LV var con .L var .LN var L(LN(	EA	AP sym DP DP *

5. Programmarchivierung

Die Einordnung von Programmen und Bausteinen in eine Programmbibliothek gewährleistet später ein einfaches Recherchieren und dient zur rationellen Programmentwicklung. Zur Effektivierung des Programmiergeschehens kommt der Fachsprache, d. h. deren Syntax und Semantik, sowie deren Implementierung zentrale Bedeutung zu. Folgende Anforderungen sind zu erfüllen:

- verständliche Notation für breite Anwenderkreise
- eindeutig, übersichtlich, problemtransparent
- konsequentes Blockkonzept
- realisierungsunabhängig
- rechnergestützte Implementierung und Testung
- Standardisierung.

Mit den Standardisierungsbestrebungen der letzten Jahre (VDI 2880, DIN 40719) wurden entsprechende Fachsprachen für SPS realisiert, z. B. DOLOG 80A für LOGISTAT AO20 (AEG) /13/, SPS600-Programmierung (Schele) /14/, STEP5 für SIMATIC S5 (Siemens) /15/. Ebenfalls unter diesen Gesichtspunkten wurde die Fachsprache MPSS (Mehr-Prozeß-Steuer-Sprache) /6/ konzipiert. Sie ermöglicht die Programmierung von Bit- und Wortverarbeitung in Anweisungslistenform. Durch das Bausteinkonzept kann das Steuerprogramm modular gestaltet werden. Darüber hinaus können Teilabläufe im Sinne von Prozessen programmiert und deren quasiparallele Ausführung mit Aktivierungs- und Deaktivierungsoperationen gesteuert werden. Auf Benennung und Priorisierung von Prozessen sowie auf beeinflußbare Warteschlangenorganisation wurde verzichtet. Das vereinfacht die Programmierung und fördert die problemtransparente Programmstrukturierung ohne Berücksichtigung von Betriebsmittelkonkurrenz. Mit MPSS wird auf assemblernahem Niveau eine virtuelle SPS programmiert.

Im Gegensatz dazu beschränkt sich die Sprache BOOL600 /16/ vorrangig auf Binäroperationen, die als Komplex Boolescher Ausdrücke die Steuerfunktion repräsentieren. Neben der unzulänglichen Ablauforientierung der Sprache sind Erweiterungen für Wortfunktionen nur unter Einbindung von Maschinenkode in die Quelle möglich.

Die MPSS-Befehle werden je nach Operandenformat, Wort oder Bit, mit dem Wort- oder Bitakkumulator der virtuellen Maschine ausgeführt. Es existieren binäre Operationen (Logik), arithmetische Operationen (Addition, Subtraktion, Multiplikation, Division, Modulo), Zähl- (vorwärts, rückwärts), Vergleichs- und Zeitoperationen (Einschalt-, Ausschaltverzögerung, Impuls). Mit Unterprogrammaufrufen sind Bausteine nutzbar, die in MPSS geschrieben sind (sog. Programmbausteine), die aber auch bereits als Maschinenprogramm, z. B. Regelfunktionen, vorliegen können (sog. Funktionsbausteine). Tafel 1 stellt den Befehlssatz im Überblick dar. In Bild 2 ist ein Beispiel für die Strukturierung eines Anwenderprogramms gegeben. Zur Realisierung von MPSS-Operationen sind je nach Peripherieausbau, z. B. bei Nutzung von Zählern und Zeitgliedern, Betriebssystemkomponenten notwendig. Die eingeschränkten Verwaltungsoperationen für parallele Prozesse erleichtern die SPS-Programmierung auf der untersten Ebene einer Automatisierungslösung wesentlich. Die Sprache zielt in erster Linie auf den SPS-Einsatz als Einzelsteuerung. Zum Anschluß von Leitsteuerungen müssen einfach handhabbare Kommunikationsbausteine zur Verfügung gestellt werden, deren Funktionen durch entsprechende Betriebssystemkomponenten (Kommunikationsprozeduren) unterstützt werden.

5. Schlußbetrachtung

Die Softwareerstellung für SPS stellt derzeit für SPS-Entwickler und -Anwender ein Hauptproblem dar. Besonders unter der Bedingung vieler eigenständiger Elektronik-Betriebe, die SPS für Maschinen- und Anlagensteuerung entwickeln, ist durch Softwareportabilität hoher ökonomischer Nutzen erzielbar. Für Anwenderprogramme ist dazu eine Softwareschnittstelle zur SPS erforderlich, die Hardware und Betriebssystem weitgehend abschirmt. Einen Ansatz in diese

(Fortsetzung auf S.24)

Grafiken über Iteration

Prof. Dr. Horst Völz, Berlin

Das Apfelmännchen ist in den letzten Jahren zum Inbegriff für vielfältige Grafiken geworden. In vielen Ausschnitten unterschiedlicher Vergrößerung zeigt es an ausgewählten Stellen immer neue und doch irgendwie gleiche bis ähnliche Strukturen, die einen beachtlichen Reiz auslösen. Hier soll nun das Grundprinzip allgemeiner aufbereitet werden. Dadurch wird angestrebt, daß der Leser selber neue Formeln und damit wiederum auch neue Strukturen findet. Vielleicht kann sich daraus so etwas wie ein Wettbewerb um die „schönsten" Bilder entwickeln. Damit die Problematik anschaulich bleibt, sind mehrere aufeinander abgestimmte Beiträge geplant. In jedem Falle wird dabei nur der KC 85/2 bzw. 3 vorausgesetzt. Für die ab dem nächsten Heft folgenden Farbbilder ist allerdings ein 64-K-RAM-Modul erforderlich. Auch die Zeiten werden dann beträchtlich; z. T. treten Rechenzeiten von über einer Woche auf. Dennoch muß betont werden, daß es eigentlich fast verwunderlich ist, daß diese Problematik überhaupt auf einem Kleinstrechner und noch dazu in BASIC behandelbar ist.

```
10 WINDOW 0,31,0,39: CLS: PRINT TAB(10);"## FEIGZ ##":PRINT
20 INPUT"Startwert =";X: A=.5
30 FOR I=0 TO 255 STEP 63.7
40  LINE 0,I,5,I,7: LINE 314,I,319,I,7
50 NEXT
60 FOR I=0 TO 319
70   A=X*A*(1-A)
80   IF A<0 THEN A=0: J=320
90   IF A>1 THEN A=1: J=320
100   PSET I,A*255,7
110 NEXT
120 INPUT"";A: IF A<0 THEN120
```

Tafel 1 ▲

Tafel 2 ▼

```
10 WINDOW 0,31,0,39: CLS: PRINT TAB(10);"## FEIGB ##":PRINT
20 INPUT"X1, X2";X1, X2: XD=(X2-X1)/319: X=X1-XD: DIM A(6)
30 FOR I=0 TO 6: PRINT I;:INPUT"-ter Wert =";A(I): NEXT
40 CLS: LOCATE 29,0:PRINTX1;" X ";X2
50 FOR I=0 TO6:J=(A(I)-X1)/XD: LINE J,0,J,5,7: NEXT
60 FOR I=0 TO 255 STEP 85
70  LINE 0,I,5,I,7: LINE 314,I,319,I,7
80 NEXT: LINE 0,0,0,255,7: LINE 319,0,319,255,7
90 PRINT AT (21,1);"0": PRINT AT (10,1);"1"
100 FOR I=0 TO 319: X=X+XD: A=.5
110  FOR J=-30 TO 100
120   A=X*A*(1-A)
130   IF A<-1 THEN A=-1: J=100
140   IF A>2  THEN A= 2: J=100
150   IF J>0  THEN PSET I,85+A*85,7
160 NEXT: NEXT
170 INPUT"";A: IF A 0 THEN170
```

Das Prinzip der Iteration

Wenn eine Gleichung in der Form
$f(X) = 0$
gegeben ist, so kann die Nullstellenbestimmung nach der Newtonschen Methode erfolgen. Es kann mit einem Startwert X0 begonnen werden. Einen besseren Wert erhält man dann, wenn man die Tangente im entsprechenden Funktionswert f(X0) bildet und damit den Schnittpunkt mit der X-Achse sucht:

$$X_{n+1} = X_n - \frac{f(X_n)}{f'(X_n)}$$

Hierbei ist mit den Indizes gleich der Iterationsprozeß allgemein nicht nur für X0 nach X1 beschrieben. Normalerweise konvergiert dieses Verfahren gegen die wirkliche Nullstelle. Es gibt aber auch Startwerte und Funktionen, bei denen der Prozeß divergiert oder andere Eigenschaften zeigt. Genau darum wird es im folgenden gehen.

Eine besonders einfache Funktion

Wir bezeichnen jetzt den Startwert mit X, und A sei ein Parameter der Funktion, über den die Iteration erfolgt. Die entscheidende Stelle des BASIC-Programms lautet dann:

```
FOR I = 0 TO B
   A = X * A * (A - 1)
NEXT
```

Es ist also zu prüfen, was bei jedem Schritt mit A geschieht. Dies ist auch deutlich von dem Startwert abhängig.
In dem Programm FEIGZ (Tafel 1) wählen wir A = .5.
Es stellt dann für 320 Iterationsschritte das jeweilige Ergebnis auf dem Bildschirm dar. Die Funktionswerte werden dabei in den Zeilen 80 und 90 auf $0 < A < 1$ begrenzt. Bild 1 zeigt die Ergebnisse für einige Startwerte. In a), also bei X = .95, konvergiert die Funktion gegen Null. Bei X = 2.98 schwankt die Funktion zunächst periodisch zwischen zwei Werten, die sich ständig nähern und auf etwa .67 einschwingen. Bereits bei X = 3.01 bleibt dieses periodische Schwanken bis in die Ewigkeit erhalten. Es erfolgt also keine Konvergenz mehr. Aber auch hier existiert ein Einschwingen. Für X = 3.399 zeigen sich beim Einschwingen zunächst 4 periodische Werte, die sich allerdings auf zwei reduzieren. Erhöht man den Startwert weiter auf 3.575, so wird selbst die Periodizität nochmals in sich periodisch. Es treten aber auch bereits gewisse Unregelmäßigkeiten auf. Bei X = 3.999 herrscht nur noch der Zufall, das Chaos. Es ist keine Vorhersage mehr möglich. Fassen wir also das Ergebnis zusammen:
Es sind zwei Zustände zu unterscheiden:
a) Der Einschwingvorgang, der unterschiedlich lange dauern kann, aber nach etwa 30 bis 100 Iterationen abgeschlossen ist.
b) Nach dem Einschwingvorgang sind vier Fälle mit Übergängen zu unterscheiden:
– Die Folge konvergiert auf einen festen Wert zu.
– Die Folge schwankt periodisch zwischen 2, 4, 8, 16 ... Werten.
– Die Folge zeigt chaotisches Verhalten.
– Die Folge divergiert, wächst über alle Grenzen.

Der Feigenbaum

Eine übersichtliche Darstellung dieser Zusammenhänge geht auf Feigenbaum zurück. Das entsprechende Programm ist FEIGB. Es wird der Startwert systematisch in einem vorzugebenden Bereich gewählt. Dann werden 30 Iterationen durchgeführt (−30 bis −1; Zeile 110 und 150). Von da ab wird jeder Iterationswert (insgesamt 100) als Punkt gesetzt. So entsteht ein Bild, dessen Abszisse die ausgewählten Startwerte enthält und dessen Ordinate die Iterationspunkte zeigt. Im Bereich von den Startwerten $-2.5 < x < 4.5$ entsteht das im Bild 2 gezeigte Ergebnis. Der Konvergenzbereich liegt danach zwischen etwa −1 und 3. Von da ab werden nach beiden Seiten zunächst 2, dann 4 und schließlich 8 periodische Werte sichtbar. Danach folgt der stochastische Bereich. Schließlich divergiert die Folge bei $X < -2$ und $X > 4$ zu extrem großen bzw. kleinen Werten. Bereits in diesem Bild ist zu erkennen, daß auch die stochastischen Bereiche gewisse Gesetzmäßigkeiten besitzen. Dies wird wesentlich deutlicher in Ausschnitten, nämlich in den Bildern 3 und 4. In Bild 3 sind auch deutlich noch Reste der Einschwingvorgänge in der Umgebung von 3 zu erkennen. Bei genauerer Be-

---

(Fortsetzung von S. 23)

Richtung stellt die Fachsprache MPSS dar. Die Anpassung von Betriebssystemsoftware auf andere Hardwarekonfigurationen ist wesentlich komplizierter. Sie wird durch Bezugnahme der Betriebssystemkomponenten auf Datenblöcke, die die SPS-Hardware beschreiben, nur in beschränkter Weise möglich. Die SPS-Programmierung wird in zunehmendem Maße mit Fachsprachen praktiziert, die an internationale Normen angelehnt sind. Dabei unterstützen und fördern Bausteine die Programmstrukturierung und -modularisierung. In Zukunft müssen Sprachmittel und Betriebssystemleistungen zur Verfügung stehen, die die Kommunikationsaufgaben zwischen SPS im Sinne eines Steuerungsverbundes lösen und die einfache Anwendung ermöglichen.

Literatur

/1/ PC-Datenbank 1984. maschine + werkzeug 5/84, S. 29 – 48
/2/ Egle, F., u. a.: SINEC-Kommunikationsverbund in der Automatisierungstechnik. Siemens Energie & Automation 7/85, H. 2, S. 165 – 168
/3/ Heger, D.: MAP-Leitlinie für die Kommunikation in der Automatisierungstechnik. Konferenzbericht 18. IPA-Arbeitstagung. Springer Verlag 1985, S. 171 – 201
/4/ Automatische Steuerung. TGL 14591 1/1978 S. 1 – 31
/5/ Brich, P.; Trapp, L.: SIMATIC-Vision, die Mensch-Maschine-Schnittstelle im Automatisierungssystem SIMATIC S5.
/6/ Programmieranleitung MRS702/703. VEB NUMERIK „Karl-Marx" Karl-Marx-Stadt
/7/ Bräuer, Ch., u. a.: PG635, PG675 und PG695: Eine neue Programmiergerätelandschaft. Siemens Energie & Automation 7/85, H. 2, S. 145 – 147
/8/ Systemunterlagendokumentation MRTS700. VEB NUMERIK „Karl Marx" Karl-Marx-Stadt 11/85
/9/ Kunke, W.; Zeidler, S.: Programmierung von Ablaufsteuerungen. Zum IEC-Standard für SPS. Mikroprozessortechnik Berlin 1 (1987) 4, S. 115 – 117
/10/ Projektierungsrichtlinie PC600. VEB NUMERIK „Karl Marx" Karl-Marx-Stadt
/12/ Herforth, M.; Petry, J.: Programmierung von SPS. etz Bd. 102 1981, H. 18, S. 963 – 968
/13/ Modulares Automatisierungssystem LOGISTAT CP80 – A500. AEG Telefunken, Prospektmaterial 1986
/14/ SPS600. Schiele, Prospektmaterial 1986
/15/ SIMATIC S5. Siemens, Katalog: ST52, Teil B, 1985
/16/ Programmieranleitung PC600. VEB NUMERIK „Karl Marx" Karl-Marx-Stadt

** FEIGZ **
Startwert = .95

ϕ.25
ϕ
a)

Startwert = 2.98

ϕ.75
ϕ.5
b)

Startwert = 3.01

ϕ.75
ϕ.5
c)

Startwert = 3.399

ϕ.75
ϕ.5
ϕ.25
d)

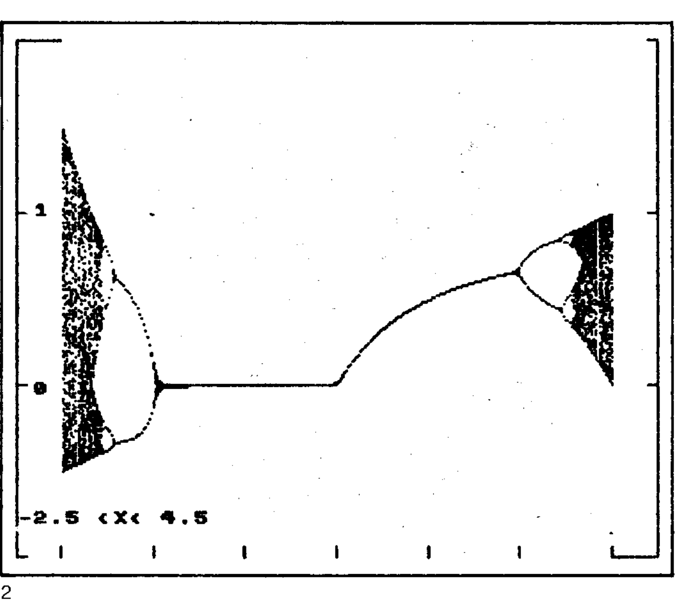
Startwert = 3.575

ϕ.75
ϕ.5
ϕ.25
e)

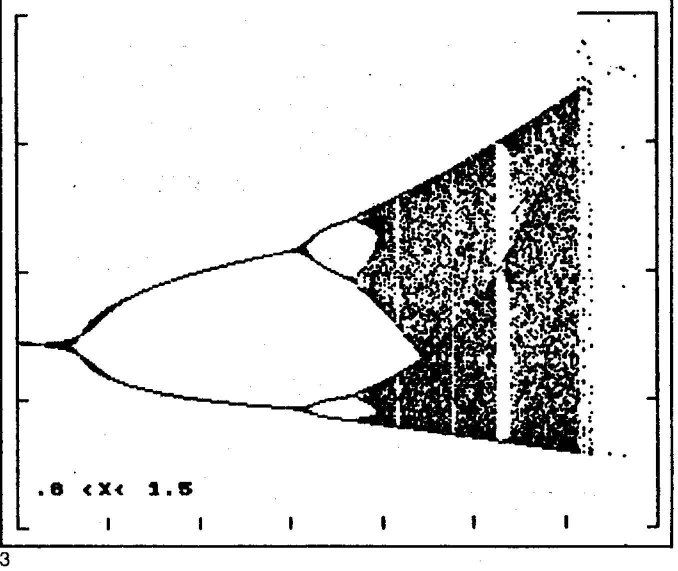
1.ϕ ** FEIGZ **
Startwert = 3.999

ϕ.75
ϕ.5
ϕ.25
ϕ
f)

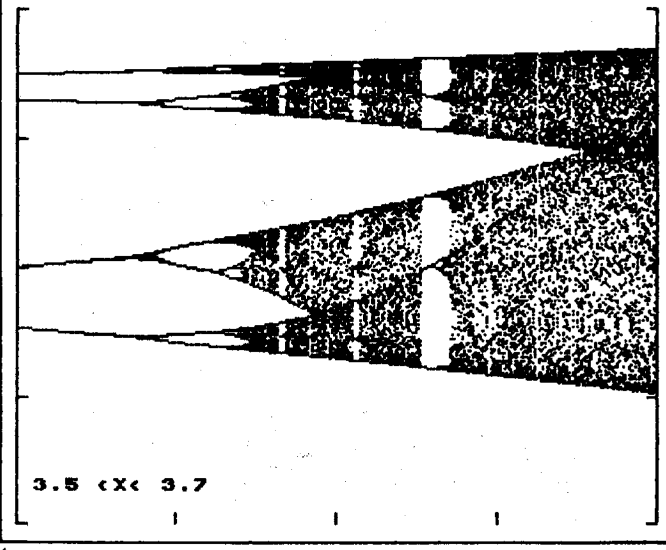
1



2



3



4

trachtung zeigt sich dann, daß die Einschwingvorgänge besonders in der Umgebung der Bifurkationspunkte lange dauern.

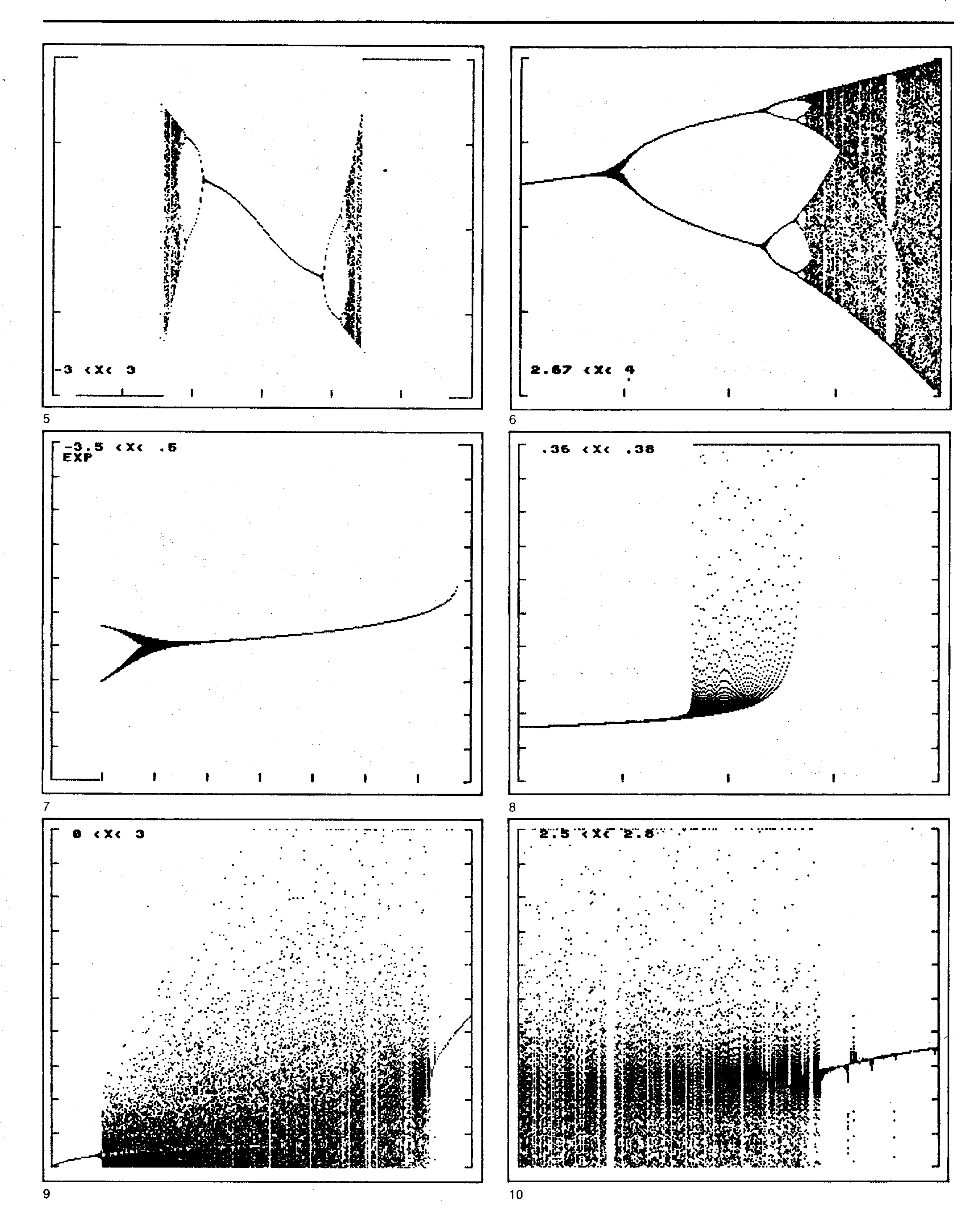
Andere Funktionen

Weitere Funktionen für Zeile 70 mit kleinen Änderungen für den Y-Bereich sind:

A: $A = X * (A * A - 1)$
B: $A = X * EXP(A)$
C: $A = X * ABS(LN(A))$

Zur Funktion A gehören die Bilder 5 und 6. Hier wird die Symmetrie der Funktion deutlich. Aber im Prinzip ist diese Funktion in ihrem Verhalten der ersten Funktion sehr ähnlich.

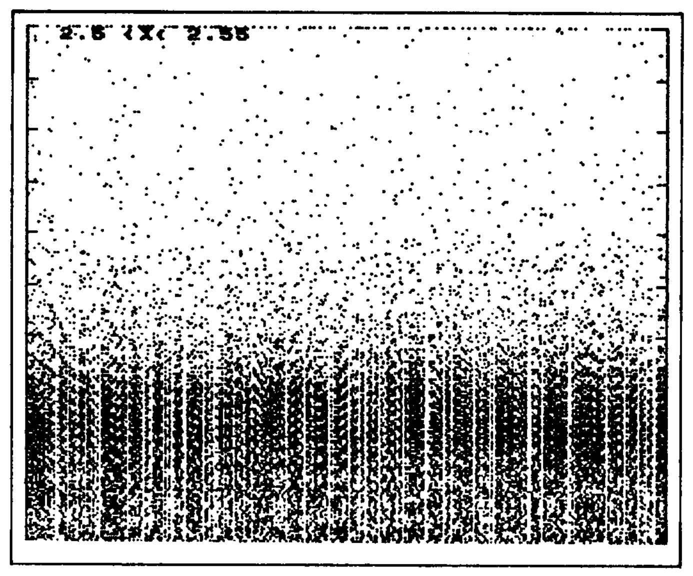
Ein erheblich anderes Verhalten zeigt die Funktion C. Sie besitzt bei kleinen Werten nur die 2fachperiodische Struktur und geht dann zu extrem kleinen Werten divergent über. Bei



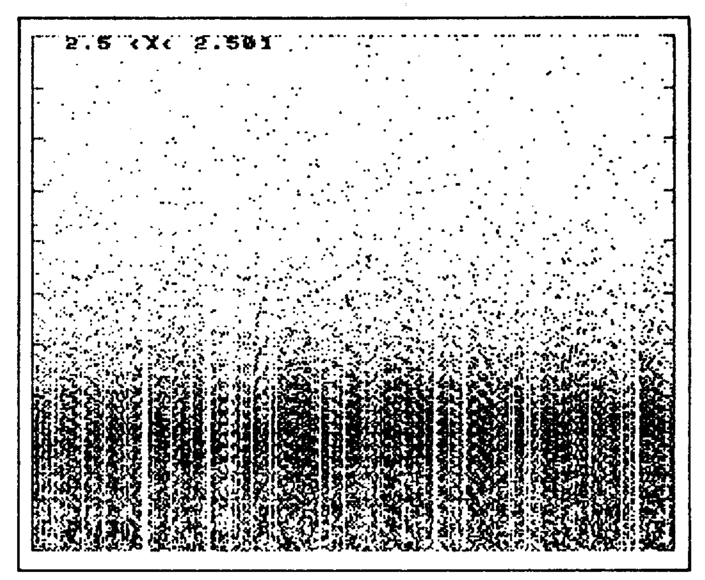
5 6 7 8 9 10

positiven Werten um .37 divergiert die Funktion gegen positiv sehr große Werte. Der Ausschnitt von Bild 8 zeigt hier den Übergang des Einschwingens. Es tritt dabei vorher keine Periodizität auf.

Als letztes Beispiel sei die Funktion C betrachtet. Sie konvergiert für Startwerte kleiner etwa .5 und größer 2.7. Dazwischen besteht im wesentlichen stochastisches Verhalten. Das Erstaunliche ist dabei, daß alle Ausschnitte in diesem Bereich immer zufällig bleiben. Dabei liegt eine bestimmte Häufigkeitsverteilung vor (Bild 12), die ein Maximum um etwa 2.5 besitzt. Nach diesen Betrachtungen dürfte es Ihnen möglicch sein, weitere Funktionen und vielleicht auch neu, interessant aussehende Feigenbaum-Diagramme zu finden.



11



12

Fraktale vom KC 85/3

Prof. Dr. Horst Völz, Berlin

Erklärung des Prinzips

Im Beitrag /1/ wurde gezeigt, wie unterschiedlich iteratives Verhalten von eindimensionalen Funktionen sein kann. Wesentlich vielfältiger sind die Möglichkeiten, wenn man nicht einen Parameter, sondern 2 verwendet. Ihre Werte sind dann in der x-y-Ebene anzuordnen. Sie entsprechen also den einzelnen Punkten auf dem Bildschirm. Beim KC 85/3 sind dies 320 ∗ 256 = 81 920 Punkte. Für jeden Punkt ist nun zu prüfen, wie er sich bei der Iteration verhält. Hier bestehen viele Möglichkeiten. Es ist bei den meisten Anwendungen jedoch nur üblich zu prüfen, ob eine gewisse Grenze G von z. B. 100 für x oder y überschritten wird. Wichtig ist dabei die Anzahl der Iterationsschritte, mit denen diese Grenze überschritten wird. Es gibt aber auch Punkte, bei denen Konvergenz auf einen festen Wert erfolgt. Weiter gibt es Punkte, bei denen sich das Iterationsergebnis periodisch verhält (vgl. /1/). In diesen Fällen wird nie die Grenze G erreicht. Deshalb muß eine Obergrenze Z für die Iterationszyklen festgelegt werden. Hierfür kann kein universeller Wert angegeben werden. In /1/ hatten wir gesehen, daß der dortige Wert 30 vielfach noch nicht ausreicht. Bei den folgenden Beispielen sind manchmal 1000 Zyklen notwendig. Wir werden später sehen, daß hier ein Kompromiß zwischen der Zeit für die Berechnung der Grafik und Auflösung der Grafik je nach dem ausgewählten Gebiet eingegangen werden muß.

Das je Punkt relevante Ergebnis kann auf unterschiedliche Weise für die bildliche Darstellung genutzt werden:

a) Man führt eine Fensterfunktion ein, die beispielsweise so aussehen könnte:
0 < rot < n1 < gelb < n2 < grün < n3 < blau < n4 < schwarz = Z

b) Man wählt die Modulofunktion. Für die 5 Werte würde dann bei der erreichten Zahl N gelten:
F = N − 5 ∗ INT (N/5)
und es würde die Farbe gemäß F = 0, 1, 2 ... 4 ausgedruckt.

Wir entscheiden uns hier für den Fall b). Er ist besonders übersichtlich.

Beschreibung der Funktion

Das sogenannte Apfelmännchen entsteht bei einer speziellen Funktion für die Iteration, die mehrfach in der Literatur beschrieben wurde:

1. Es werden Startwerte für die Iterationsgrößen A und B gesetzt. Üblich ist hier, sie gleich den Funktionswerten aus der Ebene der Pixel zu wählen, also
A = X : B = Y
Seltener werden feste Startwerte, z. B.
A = .5 : B = 0 oder A = 1 : B = 1 gewählt.
2. Die Iteration muß nun gleichzeitig für beide Größen in bezug auf die Funktionswerte der Ebene erfolgen:
$A_n A_{n-1}^2 - B_{n-1}^2 - X$
$Bn = 2 * A_{n-1} * F_{n-1} - Y$
Um mit möglichst wenig Zeitaufwand die Operationen auszulösen, ist folgende Programmierung in BASIC optimal:

```
M = A * A - B * B - X
B = B * A
B = B + B - Y
A = M
```

Zur schnellen Abwicklung der Numerik sind diese Variablen in Zeile 20 als erste initialisiert.

Besonderheiten beim KC 85/3

Die Farbgrafik des KC 85/3 ermöglicht infolge der relativ großen Farbbyte, selbst wenn man für jedes Farbbyte links die Vordergrund- und rechts die Hintergrundfarbe nutzt, maximal

80 ∗ 64 = 5120

Punkte Auflösung. Doch hier soll nun die höchstmögliche Variante beschrieben werden. Sie geht davon aus, daß die einzelnen Pixel nur schwarzweiß als je ein Farbauszug verwendet und gleichzeitig mehrere Farbauszüge berechnet und gespeichert werden. Alle Farbauszüge werden nach dem einmalig abgelaufenen Rechenvorgang gespeichert, einzeln auf dem Bildschirm erzeugt und von dort als Hardcopy mittels Nadeldrucker K 6313 ausgegeben. Danach ist es nur noch ein drucktechnisches Problem, die Farbauszüge konturendeckend zu reproduzieren. Das Ergebnis zeigen die Bilder auf den Umschlagseiten. (Dabei erfolgte auf den Umschlagseiten 3 und 4 bereits ein Vorgriff auf die folgenden Arbeiten mit neuen Funktionen.) Bei den Versuchen war es erstaunlich, mit welcher Präzision an Maßhaltigkeit der K 6313 arbeitete.

Die Speicherung des Rechenergebnisses

Das vollständige Programm ist als FAPFEL abgedruckt. Es verlangt einen 64-K-RAM-Modul und für den Drucker den V24-Modul, der auf Hardcopy initialisiert ist (z. B. im rechten Schacht 8). Der 64-K-RAM wird dann links gesteckt und mit SWITCH C 43 initialisiert. Nur unter diesen Bedingungen ist es möglich, das notwendige Feld mit DIM A (31, 319) zu dimensionieren. Entsprechend den 4 Byte je Zahl benötigt es
32 ∗ 320 ∗ 4 = 40 960 Byte.
Um aber die 5 Farbauszüge für 256 Zeilen unterzubringen, müssen folglich sogar je Zahl 5 Farben für 8 Pixel gespeichert werden (Bild 1). Dies wird über eine spezielle Codierung realisiert.
Mit den Farbwerten $F_0 \ldots F_4$, die den Pixeln 0 bis 7 zugeordnet sind, gilt für den errechneten Wert:

$$W = \sum_{i=0}^{7} F_i 5^i$$

Die größtmögliche Zahl ist dann 781 249. Die Decodierung erfolgt demgemäß rückwärts mit Division von 5^i, wobei der erste Teiler B = 78 125 beträgt (Zeile 180 im Programm). Er wird dann schrittweise mittels Division durch 5 verkleinert.

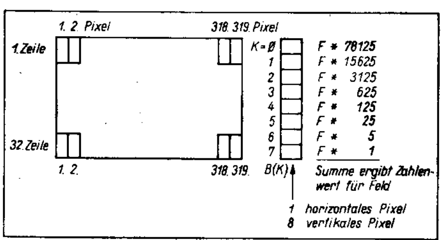


Bild 1 Codierung des Bildschirms, der Pixel und Farbwerte

Die Speicherung der Matrix erfolgt mit
CSAVE * "Name"; A;
und mit
CLOAD * "Name"; A;
kann sie wieder eingeladen werden.

Kurze Beschreibung des Programms

Das Programm FAPFEL besteht aus drei Teilen:

a) Die Zeilen 10 bis 50 dienen der Eingabe der Zahlen und des auszuwählenden XY-Bereiches.

b) In den Zeilen 60–150 erfolgt die iterative Berechnung der Pixelwerte. Zeile 100 enthält die o. g. Formel. In Zeile 110 erfolgt der Test, ob die Grenze schon erreicht wurde. Wenn dies der Fall ist, wird zunächst das Feld B (7) mit den N-Iterationszyklen für die 8 Punkte belegt. Die Zeilen 130 und 140 erzeugen dann die Belegung des Feldes A (J/8,K). Auf dem Bildschirm entsteht mit PSET zugleich das Bild für den Auszug der Farbe 4.

c) Die Routinen der Zeilen 160 bis 240 erlauben, aus dem gespeicherten Feld die einzelnen Farbauszüge sichtbar zu machen. Eine Besonderheit stellt die Zeile 160 dar. Sie greift auf die vier äußersten Ecken des Bildschirmes zu. Zum Druck sind sie für die Konturendeckung notwendig.

Rechenzeit

Die Rechenzeit des Programmteiles b) kann sehr groß sein. Bei Z = 1000 mit speziellen Ausschnitten lief der KC 85/3 manchmal 10 Tage. Die kürzesten Zeiten lagen bei ca. 20 Stunden. Der Bildaufbau eines Farbauszuges aus dem Feld dauert fast eine Stunde. Es muß hier mitgeteilt werden, daß der KC 85/3 bei mir für solche Probleme fast 1/4 Jahr pausenlos in Betrieb war. Trotz Gewitter und Stromschwankungen gab es erstaunlicherweise keinen einzigen Fehler. So stabil ist heute offensichtlich die Mikroelektronik.
Es muß aber auch erwähnt werden, daß bei diesem Problem der KC 85/3 an der Grenze seiner Leistungsfähigkeit ist. Andererseits ist zu beachten, daß 16-Bit-Technik mit Coprozessor und teilweise in Maschinensprache geschriebenen Programmen hier auch etwa eine halbe bis eine Stunde benötigt.

Eine Auswahl von Bildern

Das Apfelmännchen zeichnet sich durch fol-

Bild 2 Zusammenhang zwischen den 10 Farbbildern in Ergänzung zu Tafel 1

Tafel 1 *Daten der Bilder des Apfelmännchens*

Nr.	X1	X2	Y1	Y2	G	Z	Maßstab	Rechenzeit ca. [h]
1	−1	2,8	−1,5	1,5	1000	100	1	30
2	0,7	1	−0,3	−0,1	1000	100	15	60
3	0,74	0,79	−0,135	0,025	1000	150	26 : 76	100
4	0,74	0,77	−0,12	−0,097	300	300	150	100
5	0,741	0,749	−0,116	−0,11	300	1000	500	150
6	0,745	0,748	−0,1163	−0,114	300	500	1300	200
7	0,7445	0,7459	−0,1132	−0,11215	1000	1000	3400	150
8	0,74515	0,7455	−0,1131	−0,1129	300	1000	14000	200
9	0,74573	0,74545	−0,113018	−0,11298	300	1000	60000	150
10	0,74542	0,745445	−0,113016	−0,112999	300	1000	180000 :150000	250

```
10 WINDOW 0,31,0,39: CLS: PRINT TAB(12);"## APFEL ##": PRINT
20 DIM A(31,319), B(7): A=0: B=A: M=A: X=A: Y=A: N=A: G=1000
30 INPUT"Zyklen";Z
40 INPUT"X1, X2 =";X1,X2: XD=(X2-X1)/320: X=X1-XD
50 INPUT"Y1, Y2 =";Y1,Y2: YD=(Y2-Y1)/256: CLS
60 FOR I=0 TO 319: Y=Y1-YD: X=X+XD
70  FOR J=0 TO 255 STEP 8
80   FOR K=0 TO 7: Y=Y+YD: A=X: B=Y
90    FOR N=1 TO Z: B(K)=4
100 M=A*A-B*B-X: B=A*B: B=B+B-Y: A=M
110 IF ABS(A) + ABS(B)  G THEN B(K)=N-5*INT(N/5): NEXT K: GOTO 130
120 NEXT: NEXT
130 A=0: FOR K=0 TO 7: A=A*5+B(K): IF B(K)=4 THEN PSET I,J+K,7
140  NEXT: A(J/8,I)=A
150 NEXT: NEXT
160 PSET 0,0,7: PSET 319,0,7: PSET 0,255,7: PSET 319,255,7
170 INPUT"";A:IF A 0 THEN 170
180 B=78125: INPUT"Wert 0-4";X: CLS
190 FOR I=0 TO 319
200  FOR J=0 TO 31: L=8*J: C=B: A=A(J,I)
210   FOR K=0 TO 7
220    D=INT(A/C): A=A-D*C: C=C/5
230    IF D=X THEN PSET I,L+K,7
240 NEXT: NEXT: NEXT: GOTO 160
```

Bild 3 BASIC-Programm für die dargestellten Farbbilder und zur Abspeicherung des Feldes für die Bilder

gende Besonderheit aus: Es gibt ein Grundbild, welches die typische Form aufweist. In seinem Zentrum, das meist schwarz dargestellt wird, tritt keine Divergenz auf. Es ist im Farbbild 1 deutlich zu erkennen. Am Rand dieses Gebietes liegen vielfältig interessante Gebiete. Sie lassen, je nach vorhandener Rechnergenauigkeit, Ausschnittvergrößerungen bis zu mehreren Hunderttausend zu. Eine kleine, aber typische Auswahl zeigen hiervon die Farbbilder 2 bis 10. In Bild 2 ist die Zuordnung der Ausschnitte deutlich zu erkennen. Tafel 1 nennt die konkreten Werte. Mit Ausnahme von Farbbild 3 und 10 wurde bewußt der x-y-Maßstab proportional verändert.

Die immer wiederkehrenden Säulenstrukturen werden zuweilen Seepferdchen genannt, und die zentral einmündende Schnecke heißt dann Seepferdchental (Farbbild 7). Die sonnenförmige Struktur (Farbbild 6) wurde m. W. bisher nicht beschrieben. Ebensowenig auch die zweite Schnecke (Farbbild 9) in der Nähe des Seepferdchentals, und dort in der Nähe liegt wiederum ein ganz kleines Apfelmännchen. Solche kleinen Apfelmännchen sind übrigens sehr oft an verschiedenen Stellen vorhanden. Wenn man sich die Farbbilder 2 bis 4 genauer ansieht, kann man folgern, daß die Strukturen in Farbbild 5 bis 10 in jedem Wirbel vorhanden sind. Doch es gibt noch viele Gebiete, die es sich lohnt, weiter zu untersuchen. Dies sei dem Leser überlassen. An besonders hübschen Bildern wären wir interessiert. Es sollten dann aber unbedingt die Parameter gemäß Tafel 1 mitgeteilt werden. In zwei weiteren Beiträgen werde ich mich anderen, zum Teil noch nicht bekannten Fraktalen widmen und dabei deutlich machen, wie man sinnvoll auf der Suche nach neuen Fraktalen vorgeht.

Literatur

/1/ Völz, H.: Grafiken über Iteration. Mikroprozessortechnik, Berlin 2 (1988) 1, S. 24

50-Baud-Fernschreiber als Drucker

Manfred Klimroth, Berlin

Der vorliegende Beitrag beschreibt den Anschluß eines Fernschreibers (FS), speziell eines Blattschreibers RFT T 51, an den Atari 800XL. Bei dieser Lösung muß man allerdings einige Kompromisse in Kauf nehmen. Der Anwender muß sich je nach Gerät mit Klein- bzw. Großschrift, fehlenden Grafikzeichen, allerlei Umschrift, hohem Geräuschpegel, relativ langsamer Schreibgeschwindigkeit und großem Platzbedarf abfinden. Meist dürfte ein solcher FS trotzdem seinen Zweck erfüllen. Dem Leser soll hier eine solche Lösung vorgestellt werden. Bevor auf die Einzelheiten eingegangen wird, soll das Grundprinzip vorgestellt werden. Über den auf Ausgabe geschalteten Joystickport 1 wird dem FS durch eine Interfaceschaltung das FS-Signal zugeführt. Für dessen Erzeugung ist ein kurzes Maschinenprogramm verantwortlich. Die Bereitstellung der Impulsmuster übernimmt ein BASIC-Programm, das direkt mit der Maschinenroute kommuniziert.

Hardware

Die Schaltung ist in Bild 1 ersichtlich. Das Interface erhält von Anschluß 1 Port 1 sein Signal. Liegt dort H-Pegel an, steuert T1 durch und die LED des Optokopplers bewirkt einen Emitterstrom in dessen Fototransistor, der auch Basisstrom von T2 ist. T2 liefert dann den durch R4 und den Spulenwiderstand auf 40 mA begrenzten Kollektorstrom für die Stromspulen des Fernschreibers. Die RC-Kombination R5, C1 und C2 schützt den Transistor vor zu hohen Induktionsspannungen. Der Optokoppler verhindert, daß eventuelle Überspannungen den Computer beschädigen. Die Sendekontakte des FS sind mit in den Stromkreis einbezogen. Dies ist zwar nicht erforderlich, ermöglicht aber ein manuelles Schreiben auf der Fernschreibertastatur, wenn der Computer kein Signal sendet.

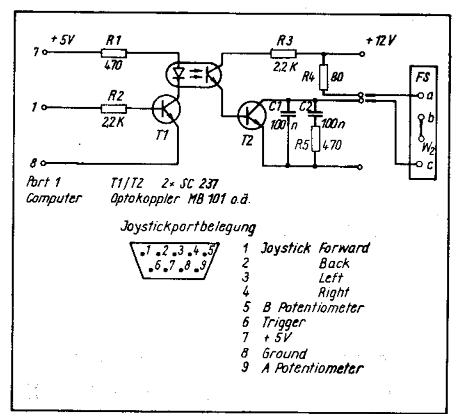


Bild 1 Interfaceschaltung

Den Spulenzweig findet man leicht mit einem Ohmmeter.

Nun noch einiges zum FS-Signal, das seriell übertragen wird. Es ist low-aktiv, d. h., bei 40 mA Dauerstrom ist der Decoder des Fernschreibers in Ruhe. Beim CCITT-Code Nr. 2 wird folgende Norm verwendet. Die Impulslänge beträgt bei 50 Baud 20 ms. Der erste Impuls ist ein Startschritt. Er ist stromlos. Die folgenden 5 Impulse stellen das Zeichen dar, und es schließen sich 1½ Stoppschritte an. Diese Stoppschritte sind Stromschritte und können beliebig verlängert werden. Beim vorliegenden Programm werden sie auf 2 Schritte gleich 40 ms ausgedehnt, was einfacher zu erreichen ist und eine höhere Datensicherheit bewirkt. Allerdings sinkt die Schreibgeschwindigkeit um 1/15. Da nur $2^5 = 32$ verschiedene FS-Codes möglich sind, ist fast jeder zweideutig. Die Eindeutigkeit wird mit Hilfe von Steuerzeichen für Buchstaben bzw. Zahlen und Zeichen wieder hergestellt. Diese Steuerzeichen bewirken ein Anheben bzw. Absenken des Fernschreiberwagens.

Software

Das FS-Signal wird von einer Routine erzeugt, die alle 20 ms durchlaufen wird. Diese Zeit läßt sich sehr leicht durch einen VBI (Vertical Blank Interrupt) erreichen. Der VBI wird während des Bildwechsels eingeleitet. Diese Möglichkeit bietet das Atari-Betriebssystem. Man muß dazu in die Adressen 546(L-Byte)

und 547(H-Byte) die Adresse der Routine ablegen. Diese muß mit JMP E45F abgeschlossen werden. Näheres findet der Leser in /1/. Der Joystickport läßt sich wie folgt auf Ausgabe schalten.

POKE 54018,48; POKE 54016,255; POKE 54018,52 bzw. LDA #$30; STA $D302; LDA #$FF; STA $D300; LDA #$34; STA $D302.

Nach Ablegen einer Zahl in 54016 erscheint diese an den Joystickports 1 und 2 jeweils an den Anschlüssen 1–4 (Bit 0–3 Port 1, 4–7 Port 2).

FS-Routine

Man stelle sich das gesamte FS-Signal (8-Schritte) als Dualzahl vor. Ein BASIC-Programm versorgt das Maschinenprogramm mit diesen Zahlen (in dezimaler Form) für die zu druckenden Zeichen.
Die Zahlen werden an einer bestimmten Speicherstelle übergeben. Die Maschinenroutine holt sich dort eine Zahl und legt sie in 54016 ab. Das FS-Interface ist an Bit 0 angeschlossen und erhält damit den ersten Impuls. Anschließend wird zur Quittierung $FF = 255 in die Übergabestelle abgelegt. Das BASIC-Programm wartet auf diese Zahl, bevor es eine neue Zahl liefert. Danach wird das Maschinenprogramm wieder verlassen. Bei den nächsten 7 Bildwechseln wird es wieder angesprungen, und es rotiert die Impulsfolge jeweils um eine Stelle nach rechts. So erscheinen alle Schritte an Bit 0 von 54016 und somit am FS. Beim danach folgenden Durchlauf wird wieder eine Zahl geholt. Wird nur eine 255 vorgefunden, wird trotzdem fortgefahren. 255 entspricht aber 8 Stromschritten, was für den FS nichts weiter als Pause bedeutet. Weitere Erläuterungen sind im Assemblerlisting zu finden.
Jeder Interessierte kann sich nun ein beliebiges BASIC-Programm zuschneiden, das die Zeichencodes für das Maschinenprogramm liefert. Die Zahlen entnehme man Tafel 1. Als Starthilfe soll hier noch ein BASIC-Programm (Bild 2) vorgestellt werden, das BASIC-Programme auslistet.
In Zeile 32000 wird mit DATA-READ-Anweisungen das Maschinenprogramm „gepokt". Zeile 32100 liest die FS-Zeichencodes in das Variablenfeld A (0...127) ein. Aus diesem werden sie dann über den ASCII-Wert des jeweiligen Zeichens ermittelt und auf der Speicherstelle 28749 an das Maschinenprogramm übergeben. Das Programm wird nach dem Eintippen mit LIST „C:" abgespeichert und jeweils, nachdem das zu druckende Programm geladen ist, mit ENTER „C:" dazugeladen. Bei dem zu listenden Programm müssen alle Zeilen ab 30000 frei sein.
Folgende Umschrift ist zu beachten:
= =
< = (
> =)
* = x
∧ = v
[= (
] =)
% = /
$ = S
" = '

Alle sonstigen Grafikzeichen usw. werden durch Zwischenräume ersetzt und können von Hand nachgetragen werden.

```
30000 rem basicprinter
30001 s=28749:rem übergabespeicher
30002 flag=0:fl1=0: fl2=0: poke 82,0
30004 trap 30005:a=a(1):if peek(28672)=1
74 then 30010
30005 gosub 32000:gosub 32100:rem maschinenpro-
gramm poken u zahlencodefeld einlesen
30006 poke s-1,8
30010 graphics 0
30011 a=usr(28716):rem initialisierung j
oystickport +vbi vektor stellen
30012 poke s, 254
30013 if peek(s)()255 then 30013
30014 poke s, 208
30020 ? :? 'ausdrucken von basicprogramm
en':? :? :?
30030 ? 'anfangszeile:',:input z
30031 ?'endzeile :',:input ze
30032 ? 'zeilenabstand:',:input n
30040 rem hauptschleife
30050 graphics 0:position 0,5
30060 list z
30070 x=0:y=6:locate 0,6,a
30080 if a=160 then if z=ze then end
30081 if a=160 then z=z+1:goto 30050
30090 locate x,y,a:if a=34 and fl 2=0 the
n fl 2=1:goto 30110
30091 if a)127 then a=a-128
30100 if a=34 and fl 2=1 then fl 2=0
30110 if fl 2=1 then 30130
30120 if a=32 then fl1=fl1+1:if fl1=2 th
en fl 1=0:goto 30200
30121 if a=32 then 30131
30130 fl 1=0
30131 gosub 30511
30140 x=x+1:if x=40 then x=0:y=y+1:poke
s,208:goto 30160
30150 goto 30090
30160 if peek (s)()255 then 30160
30161 poke s, 196
30162 if peek (s)()255 then 30162
30180 goto 30090
30200 if z=ze then 30162
30180 goto 30090
30200 if z=ze then 30300
30210 z=z+n:poke s, 208
30220 if peek (s)()255 then 30220
30230 poke s,196
30231 if peek (s)()255 then 30231
30240 goto 30050
0300 poke s, 208
30310 if peek (s)()255 then 30310
30320 poke s, 196
30330 if peek (s) (255 then 30330
30340 poke s, 254:end
0500 rem zeichenauswertung
30511 if a=36 then a=83
30512 if a=42 then a=88
30520 if a=91 then a=40
30530 if a=93 then a=41
30540 if a(65 and flag=0 then gosub 3100
0
30550 if a)64 and flag=1 then gosub 3110
0
30560 poke s,a(a)
30570 if peek(s)(255 then 30570
30580 return
31000 poke s, 246
31010 if peek (s)()255 then 31010
31020 flag=1:return
31100 poke s,254
31110 if peek(s)()255 then 31110
31120 flag=0:return
32000 restore 32010:for i=28672 to 28747
:read a:poke i,a:next i:return
32010 data 174,76,112,224,8,208,18,173,7
7,112,141,0,211,202,142,76,112,169,255,1
41,77,112,76,95,228,173
32020 data 0,211,56,106,141 ,0,211,202,20
8,2,162,8,142, 76,112,76,95,228,104,169,4
8,141,2,211,169,255,141,0
32030 data 211,169,52,141,2,211,169,255
141,0,211,169,0,141,34,2,169,112,141,35,
2,96
32100 restore 32120:dim a(127):for i=0 t
o 127:read a:a(i)=a:next i:return
32120 data 200,200,200,200,200,200,200,2
00
32130 data 200,200,200,200,200,200,200,2
00
32140 data 200,200,200,200,200,200,200,2
00
32150 data 200,200,200,200,200,200,200,2
00
32160 data 200,200,202,252,202,250,200,2
02,222,228,250,226,216,198,248,250,236,2
38,230,194,212,224,234,206,204
32170 data 240,220,216,222,252,228,242,2
00,198,242,220,210,194,218,244,232,204,2
14,222,228,248,216,240,236,238
32180 data 212,202,224,206,252,230,250,2
34,226,222,200,228,252,200,200,198,242,2
20,210,194,218,244,232,204,214
32190 data 222,228,248,216,240,236,238,2
12,202,224,206,252,230,250,234,226
32200 data 200,200,200,200,200
```

Bild 2 Basic-Programm

Tafel 1 Zuordnung des Zahlencode

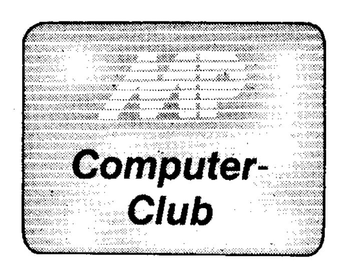
Buchstaben	*Ziff/Zei*	*Zahlencode*
A	–	*198*
B	*?*	*242*
C	*:*	*220*
D	Werda	*210*
E	*3*	*194*
F		*218*
G		*244*
H		*232*
I	*8*	*204*
J	*Wecker*	*214*
K	*(*	*222*
L	*)*	*228*
M	.	*248*
N	*,*	*216*
O	*9*	*240*
P	*0*	*236*
Q	*1*	*238*
R	*4*	*212*
S	*'*	*202*
T	*5*	*224*
U	*7*	*206*
V	=	*252*
W	*2*	*230*
X	*/*	*250*
Y	*6*	*234*
Z	+	*226*
Wagenrücklauf...		*208*
Zeilenvorschub...		*196*
A... (Buchstaben)		*254*
1... (Ziffern/Zeichen)		*246*
Zwischenraum...		*200*

Tafel 2 Assemblerlisting der Fernschreibroutine

7000	LDX	$704C	;Zähler-Impulse des Zeichens
7003	CPX	#$08	;x=8 bedeutet neues Zeichen
7005	BNE	$12	;X≠8 Sprung zu 7019
7007	LDA	$704D	;neuen Zeichencode laden
700A	STA	$D300	;Code auf Port legen
700D	DEX		;Zähler – 1
700E	STX	$704C	;speichern für nächsten Durchlauf
7011	LDA	#$FF	;Quittierung
7013	STA	$704D	;in Übergabebyte ablegen
7016	JMP	$E45F	;Rücksprung ins Betriebssystem
7019	LDA	$D300	Joyport laden
701C	SEC		;Carry setzen, wird bei ROR eingeschoben
701D	ROR		;nach rechts rot., nächster Imp. an Bit 0
701E	STA	$D300	;auf Joyport legen
7021	DEX		;Zähler – 1
7022	BNE	$02	wenn x ≠ 0 nach 7026
7024	LDX	#$08	;Zeichen ist ausgegeben; neues Zeichen beim nächsten Durchlauf
7026	STX	$704C	;Zähler ablegen
7029	JMP	$E45F	;Rücksprung

Literatur

/1/ Wiethoff, A.; Reschke, J.: Das Atari Profibuch. Düsseldorf: Sybex-Verlag 1985



KC 85/3-Assemblertip
Markenanzeige

Bei der Arbeit mit dem Modul M 027 Development /1/ zeigte sich beim Testen der mit dem Assembler erstellten Programme, daß auf ein häufiges Ausdrucken des Listings verzichtet werden kann, wenn die Adressen der im Quellprogramm benutzten Marken angezeigt werden können.
Bild 1 zeigt das Listing eines mit dem Modul M 027 erstellten Assemblerprogrammes, das sowohl die Ausgabe der Adresse einer Marke (Label) als auch die Anzeige bzw. den Ausdruck einer kompletten Markentabelle eines vom Assembler übersetzten, im Speicher stehenden Quellprogrammes. Beide Programme sind Ergänzungen des TEMO-Untermenüs. Das TEMO-Menü arbeitet mit dem Prologbyte FDH, vgl. /2/. Der Suchbereich des TEMO-Menüs erstreckt sich, wie der des EDAS-Menüs auch über den Bereich von BA00H bis BFFFH.
Nach der Assemblierung eines Quellprogrammes wird der Assembler mit der Anweisung **EXIT** verlassen. Anschließend kann durch Eingabe von **TEMO** in das Untermenü des Testmonitors übergegangen werden. Hier sind nun die Anweisungen **LABLIST** und **LABEL** verfügbar. Durch die Eingabe von **LABLIST** werden alle im Programm verwendeten Marken in der Reihenfolge ihres Auftretens im Quellprogramm mit vorangestellter Adresse auf Bildschirm oder auch zusätzlich auf Drucker ausgegeben.
Der Aufruf der Menüanweisung **LABEL** erfolgt durch Eingabe von **LABEL** ohne Parameter. Anschließend wird der Name am Anfang der folgenden Bildschirmzeile angefordert. Ist die Marke vorhanden, so erfolgt die Adressenausgabe, ansonsten die Fehlermeldung **NOT FOUND**.

Bild 1 Assemblerlisting zum Markenanzeigeprogramm

```
+LABEL
LL7
BF9C
+LABEL
MM1
NOT FOUND
+LABLIST 5
BFBD  LLE    BF9D  LL7    BF8C  LL3    BF7B  LL1    BF77  LL2
BF74  LL6    BF5D  LL0    BF4D  LA2    BF49  LA4    BF43  LA6
BF38  LA7    BF25  LA3    BF24  LA0    BF17  LA1    BF16  LA5
B7A0  CURSO  B782  ARG1   B781  ARGN   0072  MBOT   0074  MTOP
F003  CAOS
+
```

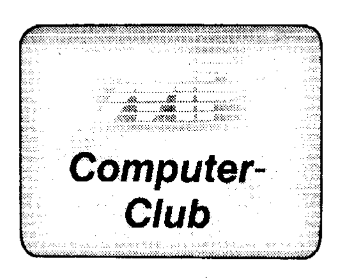
Bild 2 Beispielein- und -ausgaben mit dem Programm

1

```
BF5A 00                 DEFB    0
BF5B 18E6               JR      LA6             ;beenden
BF5D            ;AUSGABE LABEL-LISTE
BF5D 3282B7     LL0:    LD      (ARG1),A        ;Eintrag je Zeile
BF60 1812               JR      LL6
BF62            ;
BF62 FDFD               DEFW    0FDFDH
BF64 4C41424C           DEFM    'LABLIST'
BF6B 01                 DEFB    1
BF6C 3A81B7             LD      A,(ARGN)        ;Argument eingeg.?
BF6F A7                 AND     A
BF70 3E03               LD      A,3             ;wenn nein  3 Marken
BF72 28E9               JR      Z,LL0           ;je Zeile
BF74 2A7400     LL6:    LD      HL,(MTOP)
BF77 3A82B7     LL2:    LD      A,(ARG1)        ;Marken je Zeile
BF7A 47                 LD      B,A
BF7B 5E         LL1:    LD      E,M             ;Adresse low
BF7C 23                 INC     HL
BF7D 56                 LD      D,M             ;        high
BF7E 23                 INC     HL
BF7F 23                 INC     HL              ;ueberlesen 0AH
BF80 EB                 EX      DE,HL
BF81 CD03F0             CALL    CAOS
BF84 1A                 DEFB    1AH             ;HL-HEX
BF85 CD03F0             CALL    CAOS
BF88 2B                 DEFB    2BH             ;SPACE
BF89 0E00               LD      C,0             ;Markenlaenge=0
BF8B EB                 EX      DE,HL
BF8C 7E         LL3:    LD      A,M
BF8D CD03F0             CALL    CAOS
BF90 24                 DEFB    24H             ;ASCII-OUT
BF91 0C                 INC     C               ;Markenlaenge+1
BF92 23                 INC     HL
BF93 7E                 LD      A,M
BF94 FE3A               CP      ':'             ;Markenende?
BF96 20F4               JR      NZ,LL3
BF98 23                 INC     HL
BF99 3E07               LD      A,7
BF9B 91                 SUB     C               ;Ausgabe SPACE
BF9C 4F                 LD      C,A             ;bis Position
BF9D CD03F0     LL7:    CALL    CAOS            ;Markenanfang +7
BFA0 2B                 DEFB    2BH             ;SPACE
BFA1 0D                 DEC     C
BFA2 20F9               JR      NZ,LL7
BFA4 A7                 AND     A
BFA5 E5                 PUSH    HL
BFA6 C5                 PUSH    BC
BFA7 ED4B7200           LD      BC,(MBOT)       ;Markentabellenende
BFAB ED42               SBC     HL,BC           ;erreicht?
BFAD C1                 POP     BC
BFAE E1                 POP     HL
BFAF 300C               JR      NC,LLE          ;->nein
BFB1 10C8               DJNZ    LL1             ;Schleife fuer Zeile
BFB3 CD03F0             CALL    CAOS
BFB6 2C                 DEFB    2CH             ;Newline
BFB7 CD03F0             CALL    CAOS
BFBA 2A                 DEFB    2AH             ;BREAK-Test
BFBB 30BA               JR      NC,LL2          ;->nein
BFBD CD03F0     LLE:    CALL    CAOS
BFC0 2C                 DEFB    2CH             ;Newline
BFC1 C9                 RET                     ;Ende
ERRORS: 0000
```

2

```
3000            ;HILFSPROGRAMM fuer TEMO
3000            ; MARKENANZEIGE
3000            ; Created by K.-D. KIRVES
3000            ;       MPM
3000            CAOS    EQU     0F003H
3000            MTOP    EQU     74H     ;MARKENANFANG
3000            MBOT    EQU     72H     ;MARKENENDE
3000            ARGN    EQU     0B781H  ;ARGUMENTANZAHL
3000            ARG1    EQU     ARGN+1  ;ARGUMENT 1
3000            ; Anzeige einer Marke
3000                    ORG     0BF00H
BF00 FDFD               DEFW    0FDFDH  ;Prolog TEMO
BF02 4C414245           DEFM    'LABEL'
BF07 01                 DEFB    1
BF08 CD03F0             CALL    CAOS
BF0B 17                 DEFB    17H     ;LINE INPUT
BF0C 2A7400             LD      HL,(MTOP)
BF0F ED4B7200           LD      BC,(MBOT)
BF13 D5                 PUSH    DE
BF14 180E               JR      LA0
BF16            ;
BF16 D5         LA5:    PUSH    DE
BF17 7E         LA1:    LD      A,M     ;aus Markenspeicher
BF18 23                 INC     HL
BF19 E5                 PUSH    HL
BF1A A7                 AND     A
BF1B ED42               SBC     HL,BC   ;Test auf Ende
BF1D E1                 POP     HL
BF1E 302D               JR      NC,LA2
BF20 FE0A               CP      0AH     ;Trennzeichen nach
BF22 20F3               JR      NZ,LA1  ;Adresse ?
BF24 E5         LA0:    PUSH    HL
BF25 1A         LA3:    LD      A,(DE)  ;Vergleichskette
BF26 BE                 CP      M
BF27 2020               JR      NZ,LA4
BF29 23                 INC     HL
BF2A 13                 INC     DE
BF2B 7E                 LD      A,M
BF2C FE3A               CP      ':'     ;Trennzeichen nach
BF2E 20F5               JR      NZ,LA3  ;Marke ?
BF30 1A                 LD      A,(DE)  ;Ende der
BF31 A7                 AND     A       ;Vergleichskette
BF32 2804               JR      Z,LA7   ; 00 oder Space
BF34 FE20               CP      20H
BF36 2011               JR      NZ,LA4
BF38 E1         LA7:    POP     HL
BF39 2B                 DEC     HL
BF3A 2B                 DEC     HL
BF3B 5E                 LD      E,M     ;Adresse low
BF3C 2B                 DEC     HL
BF3D 6E                 LD      L,M     ;        high
BF3E 63                 LD      H,E
BF3F CD03F0             CALL    CAOS
BF42 1A                 DEFB    1AH     ;Ausgabe HL hex
BF43 D1         LA6:    POP     DE
BF44 CD03F0             CALL    CAOS
BF47 2C                 DEFB    2CH     ;Ausgabe Newline
BF48 C9                 RET
BF49 E1         LA4:    POP     HL
BF4A D1                 POP     DE
BF4B 18C9               JR      LA5     ;weiter suchen
BF4D CD03F0     LA2:    CALL    CAOS
BF50 23                 DEFB    23H
BF51 4E4F5420           DEFM    'NOT FOUND'
```



In Bild 2 wird die Arbeit mit beiden Anweisungen anhand einiger Ein- und Ausgaben dargestellt. Als Quellprogramm dient das Programm selbst.
Das Programm ist auch auf dem KC85/2 lauffähig.

K.-D. Kirves

Literatur

/1/ Kirves, K.-D.: Modul M 027 Development-Assemblerprogrammierung für KC 85/3. Mikroprozessortechnik, Berlin 1 (1987) 8, S. 247–249

/2/ Domschke, W.: Das Softwarekonzept des KC 85/3. Mikroprozessortechnik, Berlin 1 (1987) 3, S. 89–91

KC-Tip

Das Programm zeigt, wie viele Eingaben in Spaltenform möglich sind. Der aufwendigere Weg über die INKEY-Funktion entfällt.

Peter Zehrt

```
10 PRINT " FAKTOR 1"; TAB (12) "FAKTOR 2";
TAB (28) "PRODUKT"
20 WINDOW 2,23,1,10
30 FOR I = 1 TO 5: INPUT" "; F1 (I): NEXT
40 WINDOW 2,23,11,20
50 FOR I = 1 TO 5: INPUT" ";F2 (I):NEXT
60 WINDOW 2,23,27,39
70 FOR I = 1 TO 5: PRINT F1 (I) * F2 (I): NEXT
80 WINDOW
```

Direkteingabe von Funktionen in BASIC-Programme

Beim Berechnen oder Zeichnen mathematischer Funktionen mit Hilfe des Computers erhöht sich der Dialogkomfort, wenn vom Nutzer die zu verarbeitenden Funktionen mit Hilfe des INPUT-Befehls eingegeben werden können. Mit dem folgenden Unterprogramm ist das möglich.
Dabei sollte Zeile 30 in der Hintergrundfarbe geschrieben werden, um den Nutzer nicht zu verwirren. Die Speicherstelle der zuletzt gedrückten Taste erhält den Tastaturcode der RETURN-Taste.
Nach Abschluß der INPUT-Anweisung muß der Anwender noch einmal RETURN drükken.

Kay Rohnke

Beispielprogramm (Atari-BASIC)

```
:
:
10 ?CHR$(125):DIM A$(40)
20 INPUT A$
30POSITION 2,2:?"Zeilnr.";A$;":RETURN
":?"CONT"
40 POSITION 2,0:POKE 764,12
50 STOP
60 RETURN
:
:
```

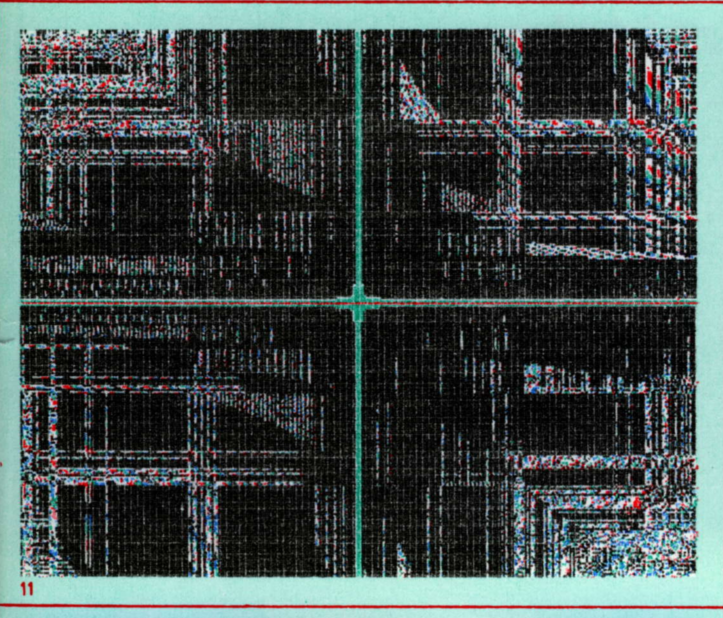
Dateiorganisation mit M 011 auf KC 85/2(/3)

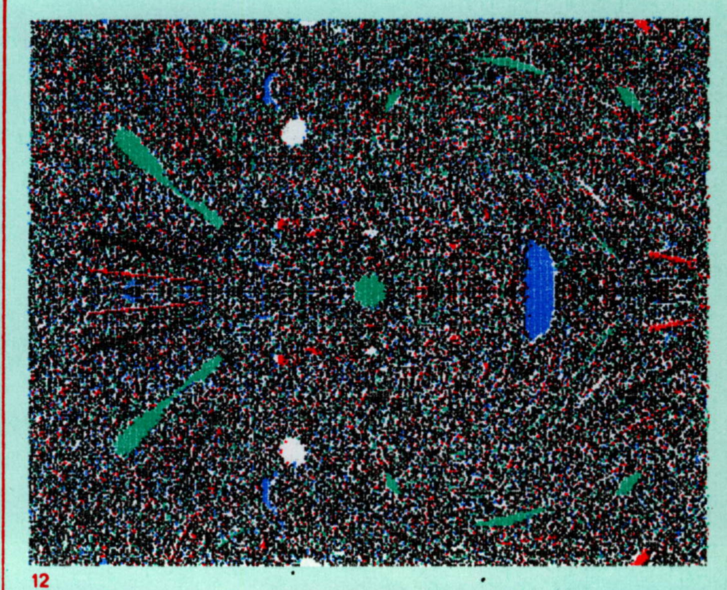
Uwe Zierott, Lehnin

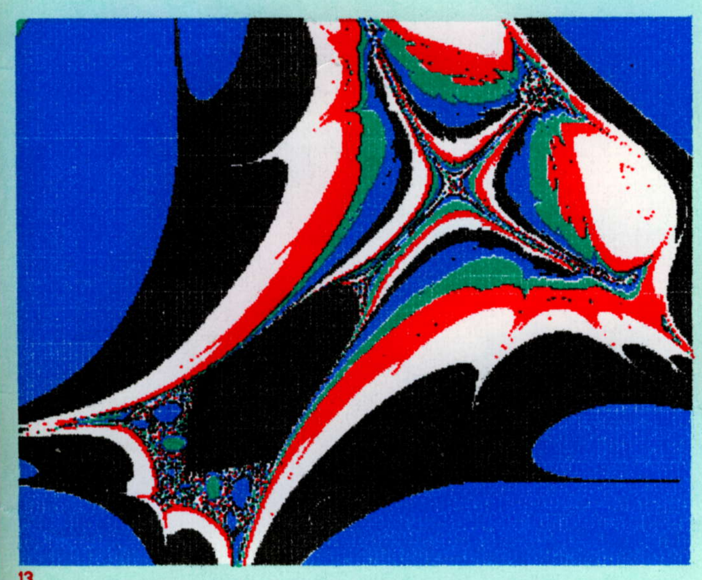
```
6000                 ORG     0BF00H
BF00        MSSP     EQU     0B800H
BF00        EBENE    EQU     MSSP+09H
BF00        BLOCK    EQU     MSSP+0AH
BF00        LAUF     EQU     MSSP+0BH
BF00        DATEI    EQU     MSSP+0DH
BF00        EBENE1   EQU     03H
BF00        EBENE2   EQU     07H
BF00        TBYTE    EQU     07FFFH   ;Testbyte
BF00        ;
BF00        ;*************************************
BF00        ;*   Schaltsoftware für alle im KC-   *
BF00        ;*   System befindlichen RAM-Module   *
BF00        ;*   mit dem Strukturbyte 0F6H        *
BF00        ;*   -----------------------------    *
BF00        ;* Die Aufteilung  erfolgt in Dateien *
BF00        ;* zu je 32Kbyte     (00000H-07FFFH). *
BF00        ;* Der Stack und die Interrupttabel-  *
BF00        ;* len werden kurzzeitig in den Kas-  *
BF00        ;* settenpuffer verlagert und auf den *
BF00        ;* aktuellen Speicherblock 1 kopiert. *
BF00        ;* Dieses Dienstprogramm ist nur im   *
BF00        ;* Speicherbereich >07FFFH lauffähig  *
BF00        ;* und kann sowohl vom CAOS- Menü als *
BF00        ;* auch vom laufenden Programm auf-   *
BF00        ;* gerufen werden. Die Dateinummer    *
BF00        ;* ist hexadezimal dem Register L zu  *
BF00        ;* übergeben. Die Schaltsoftware eig- *
BF00        ;* net sich sich für die Datenverwal- *
BF00        ;* tung in speziellen Anwenderpro-    *
BF00        ;* grammen, aber auch zur effektiven  *
BF00        ;* Programm- und Dateierstellung mit  *
BF00        ;* den Modulen M027 Development und   *
BF00        ;* M012 Texor. BASIC- Programme las-  *
BF00        ;* sen sich unabhangig voneinander    *
BF00        ;* verwalten oder durch geeignete Be- *
BF00        ;* fehlsfolgen verketten.             *
BF00        ;* Die Steuerbytes werden nicht im    *
BF00        ;* Modulsteuerspeicher  notiert.      *
BF00        ;* ---------------------------------- *
BF00        ;* In der Prioritätskette darf vor    *
BF00        ;* den M011-Modulen kein M022 im Be-  *
BF00        ;* reich 04000H-07FFFH aktiv sein.    *
BF00        ;* ---------------------------------- *
BF00        ;*   Berücksichtigt wurde auch der    *
BF00        ;*   auf 256Kbyte umgerüstete M011.   *
BF00        ;*   Mit ihm sind zusätzlich je 8     *
BF00        ;*   Dateien zu 32k realisierbar.     *
BF00        ;*   Die Reihenfolge der kontaktier-  *
BF00        ;*   ten M011 und der 256K-RAM ist    *
BF00        ;*   beliebig.                        *
BF00        ;* ---------------------------------- *
BF00        ;* Copyright by Uwe Zierott  / Lehnin *
BF00        ;*          Stand: 25.8.1987          *
BF00        ;*************************************
BF00        ;
BF00 7F7F            DEFW    07F7FH
BF02 44415445        DEFM    'DATEI
BF07 01              DEFB    1
BF08        ;PE: Dateinummer in L
BF08 FDE5   SCHALT   PUSH    IY
BF0A FD21FF7F        LD      IY,TBYTE
BF0E E5              PUSH    HL
BF0F        ;Systemarbeitszellen in den IRM
BF0F F3              DI
BF10 21FF01          LD      HL,01FFH
BF13 117FB7          LD      DE,0B77FH
BF16 018000          LD      BC,080H
BF19 EDB8            LDDR
BF1B 2180B5          LD      HL,0B580H
BF1E 39              ADD     HL,SP
BF1F F9              LD      SP,HL
BF20
BF20 E1              POP     HL
BF21 AF              XOR     A
BF22 320BB8          LD      (LAUF),A
BF25 7D              LD      A,L
BF26 320DB8          LD      (DATEI),A
BF29 CB3F            SRL     A
BF2B 3209B8          LD      (EBENE),A
BF2E FE00            CP      0
BF30 280C            JR      Z,MODAUS
BF32 CB45            BIT     0,L
BF34 1EFF            LD      E,0FFH   ;Block 1+2
BF36 2802            JR      Z,MODU1
BF38 1E7F            LD      E,07FH   ;Block 3+4
BF3A 210AB8 MODU1    LD      HL,BLOCK
BF3D 73              LD      M,E
BF3E        ;
BF3E 063E   MODAUS   LD      B,62
BF40 2E08            LD      L,08H
BF42 C5     MODUL    PUSH    BC
BF43 45              LD      B,L
BF44 0E80            LD      C,080H
BF46 ED60            IN      H,(C)
BF48 3EF6            LD      A,0F6H
BF4A BC              CP      H
BF4B 204E            JR      NZ,MODUL2
BF4D 1603            LD      D,EBENE1
BF4F ED51            OUT     (C),D
BF51 FD7E00          LD      A,(IY)
BF54 5F              LD      E,A
BF55 1607            LD      D,EBENE2
BF57 ED51            OUT     (C),D
BF59 FDBE00          CP      (IY)        ;Ebenen sind
BF5C 2012            JR      NZ,MODUL1   ;ungleich !
BF5E 3C              INC     A
BF5F FD3400          INC     (IY)
BF62 1603            LD      D,EBENE1
BF64 ED51            OUT     (C),D
BF66 FDBE00          CP      (IY)        ;schaltbar ?
BF69 1607            LD      D,EBENE2
BF6B ED51            OUT     (C),D
BF6D FD7300          LD      (IY),E
BF70 0601   MODUL1   LD      B,1
BF72 3A09B8          LD      A,(EBENE)
BF75 57              LD      D,A
BF76 3A0BB8          LD      A,(LAUF)
BF79 2003            JR      NZ,MOD256
BF7B 3C              INC     A
BF7C 1808            JR      FERTIG
BF7E        ;4 Ebenen pruefen
BF7E 0604   MOD256   LD      B,4
BF80 3C     MODU2    INC     A
BF81 BA              CP      D
BF82 2802            JR      Z,FERTIG
BF84 10FA            DJNZ    MODU2
BF86 320BB8 FERTIG   LD      (LAUF),A
BF89 BA              CP      D
BF8A 200A            JR      NZ,MODUL0
BF8C 3A0AB8          LD      A,(BLOCK)
BF8F C604   MODU3    ADD     4
BF91 10FC            DJNZ    MODU3
BF93 57              LD      D,A
BF94 1802            JR      MODEIN
BF96        ;Ebene nicht im Modul
BF96 1600   MODUL0   LD      D,0
BF98        ;aktuelles Modul schalten
BF98 45     MODEIN   LD      B,L
BF99 ED51            OUT     (C),D
BF9B 3E04   MODUL2   LD      A,4
BF9D 85              ADD     L
BF9E 6F              LD      L,A
BF9F C1              POP     BC
BFA0 10A0            DJNZ    MODUL
BFA2 3A09B8          LD      A,(EBENE)
BFA5 B7              OR      A
BFA6 2811            JR      Z,MODUL3
BFA8 DB88            IN      A,(088H)
BFAA CB8F            RES     1,A
BFAC D388            OUT     (088H),A
BFAE DD7E00          LD      A,(IX)
BFB1 DD3400          INC     (IX)
BFB4 DDBE00          CP      (IX)
BFB7 2006            JR      NZ,MODUL4
BFB9 DB88   MODUL3   IN      A,(088H)
BFBB CBCF            SET     1,A
BFBD D388            OUT     (088H),A
BFBF        ;Systemarbeitszellen in den RAM
BFBF 217FB7 MODUL4   LD      HL,0B77FH
BFC2 11FF01          LD      DE,01FFH
BFC5 018000          LD      BC,080H
BFC8 EDB8            LDDR
BFCA 21804A          LD      HL,-0B580H
BFCD 39              ADD     HL,SP
BFCE F9              LD      SP,HL
BFCF FB              EI
BFD0 FDE1            POP     IY
BFD2 C9              RET

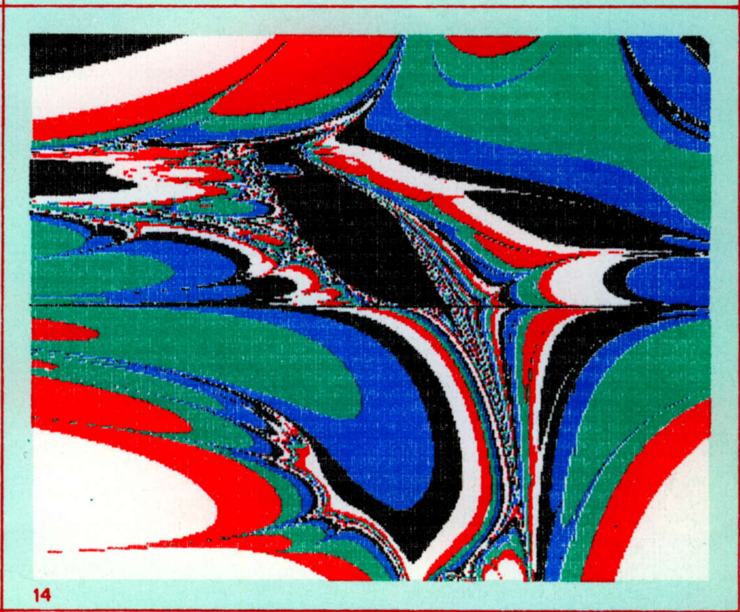
ERRORS: 0000
```

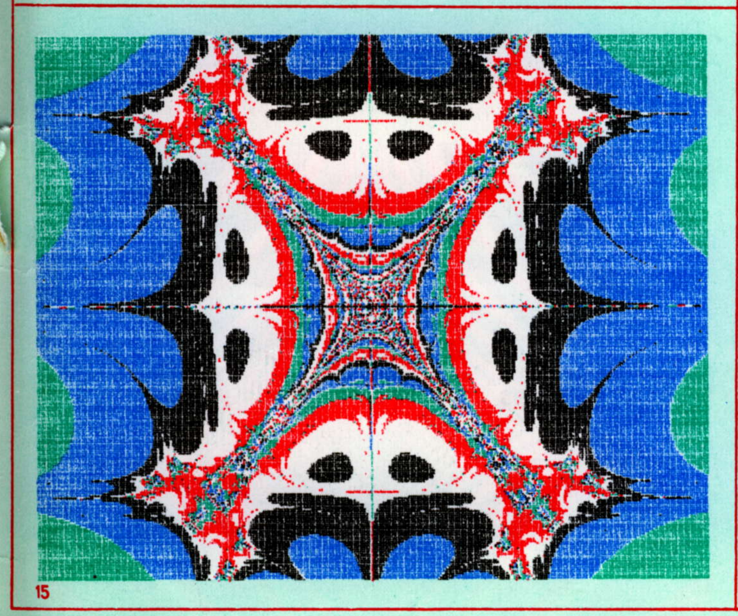
Neue Fraktale

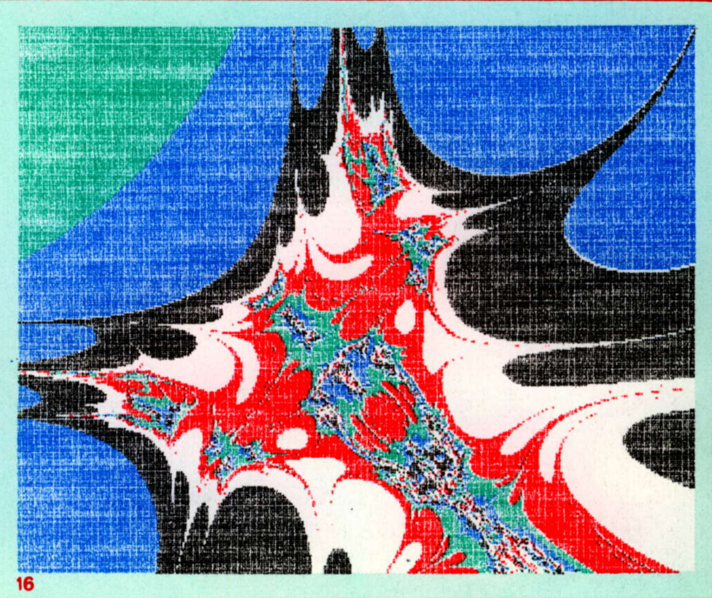

11


12


13

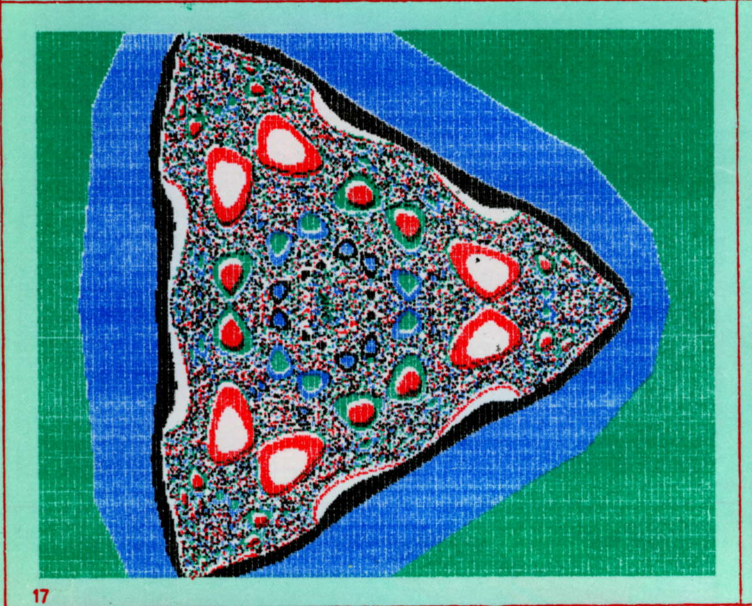

14

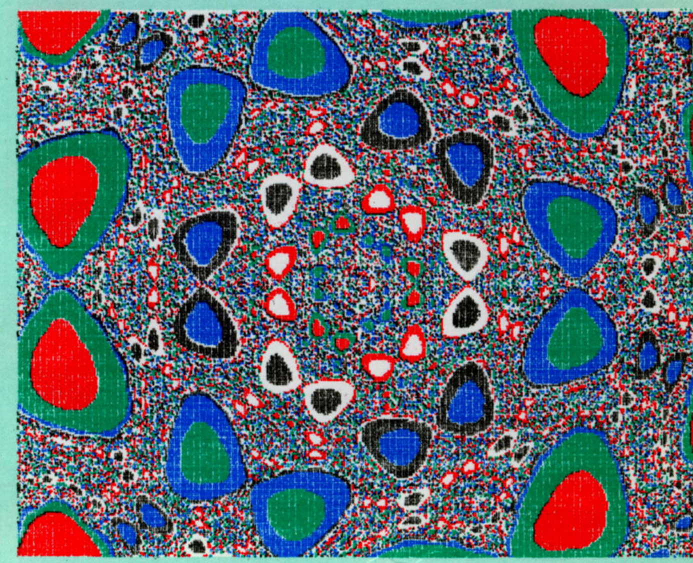

15

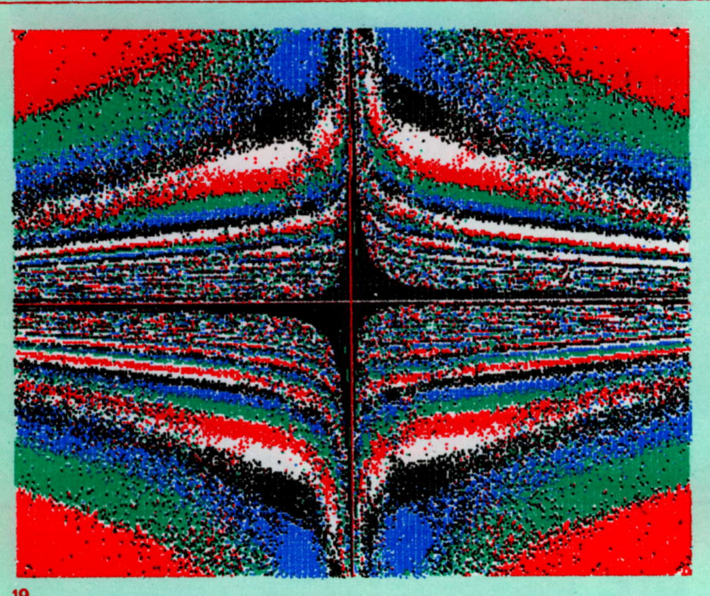

16

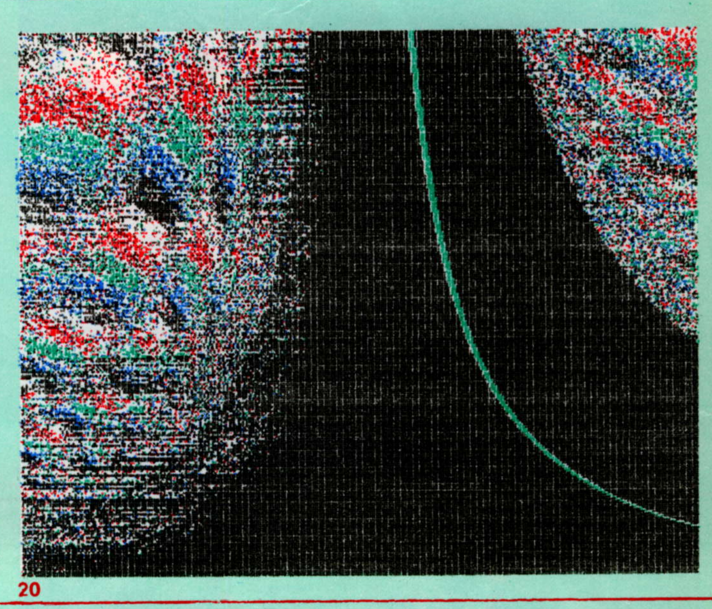
ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 1
S. 1–32

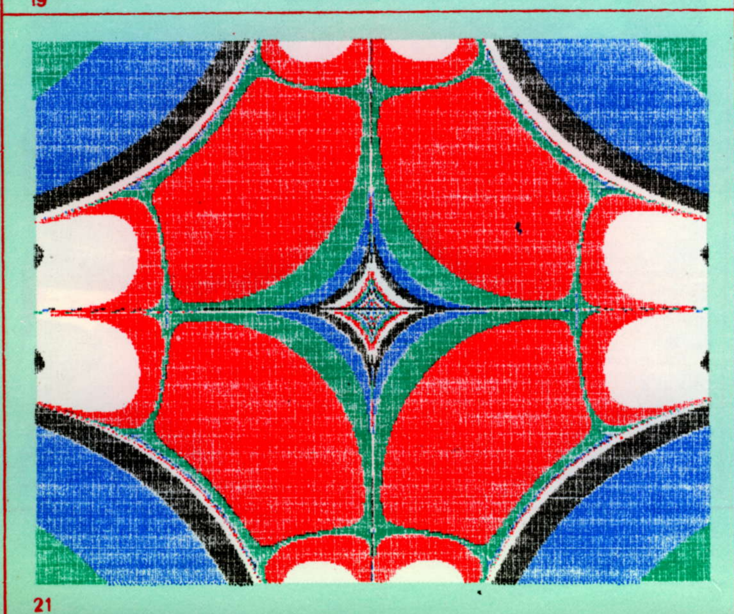
Neue Fraktale

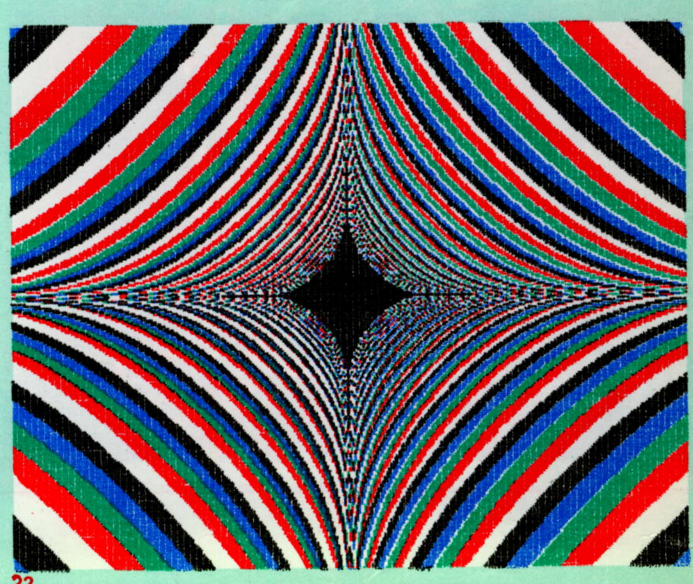

17


18


19


20


21


22



ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK
MP
Grund
Heft 2 · 1988
Mikroprozessortechnik
VEB Verlag Technik Berlin
ISSN 0232-2892
MP-Kurs:
Mikroprozessorsystem
K 1810 WM86
Neues über FORTH



Dialog

CP/M oder UDOS?

Die Meinung von A. Hoklas in MP 10/87, S. 290 ist aus der Sicht derjenigen, welche hauptsächlich mit UDOS arbeiten und in der Regel prozeßnahe Software entwickeln, sicherlich weit verbreitet und hat auch ihre Berechtigung. Heutzutage ist aber die Masse der Computernutzer mit der Lösung allgemeiner, meist ökonomischer Probleme beschäftigt. Deshalb ist meiner Meinung nach die Stellungnahme der Redaktion zu global und nicht überzeugend, so daß Neulinge auf der Computerszene tatsächlich ins Grübeln kommen könnten. UDOS wurde speziell für den Prozessor Z80 entwickelt, nutzt also auch alle Möglichkeiten dieses Prozessors. Dagegen ist CP/M, wenn man nur die Arbeitsweise des Systems betrachtet, natürlich moralisch verschlissen.
Hoch aktuell ist jedoch nach wie vor die gesamte unter CP/M verfügbare Anwendersoftware, welche den meisten Anwenderwünschen bereits ohne Benutzung einer höheren Programmiersprache gerecht wird. Gibt es denn unter UDOS etwas Vergleichbares zu WSTAR, dBASE, SUPERCALC, MULTIPLAN usw.? Vor einigen Jahren waren die Computernutzer auf Grund der geringen Verfügbarkeit der Gerätetechnik in der Regel Computerexperten. Für diese „Insider" war UDOS oder RIO, z. B. auf dem MRES, ausgesprochen interessant (Erfahrungswert). Heute hingegen gibt es bereits wesentlich mehr Computer als Experten, d. h., die stetig wachsende Masse der Nutzer sind die sogenannten „naiven" Anwender. Und für diese Nutzer ist CP/M mit seiner anwenderfreundlichen Benutzeroberfläche und seiner weitreichenden Anwendersoftware sicherlich besser geeignet als das offene UDOS mit seinen für „Outsider" recht schwer durchschaubaren Möglichkeiten.
Das Problem der Einbindung zusätzlicher Treiber in ein CP/M-kompatibles System ist für einen CP/M-Systemprogrammierer ebensowenig ein Problem, wie für einen UDOS-Systemprogrammierer die Treibereinbindung in UDOS. Auch unter UDOS gibt es einige allgemeingültige Forderungen an ein Treiberprogramm, die beachtet werden müssen. Dies ist also auch kein Argument, um gleich CP/M zu „verdammen". Die angeführten „schwerwiegenden konzeptionellen Schwächen" von CP/M entfallen weitestgehend, wenn der CP/M-Systemprogrammierer seine zusätzlichen Treiber (falls überhaupt nötig) direkt ins BIOS einlagern kann. Warum also nicht CP/M?

Hartmut Schreiber

Entwicklung Integrierter Schaltungen bis zum Jahre 2000

In dem Beitrag in MP 7/87, S. 195–200 wurde dargestellt, wohin die Mikroelektronik bis zum Jahre 2000 marschieren wird. Dennoch gibt es aus meiner Sicht einige Bemerkungen:

- Die Hauptanwendungsgebiete der Bauelemente werden sich anders verteilen als in Tafel 1 dargestellt. Beispielsweise gibt der Zweigverband der Elektrotechnik (ZVEJ) in der BRD für 1986 an: 11,6 Mrd DM Bauelemente, davon 25 % Industrieelektronik; 22 % Telekommunikation; 20 % Unterhaltungselektronik, 17 % Büro- und Datentechnik; 12 % Autoelektronik mit etwa gleichen jährlichen Zuwachsraten.
- ASICs werden nicht „einige 10 %", sondern 30 % des Weltmarktes erreichen und gravierend das Niveau bestimmen.
- Der Siegeszug der schnellen CMOS-Technik ist nicht aufzuhalten; dem sollte auch in einem solch prinzipiellen Artikel Rechnung getragen werden, bevor man Supercomputer mit ECL und GaAs besonders betont.
- Bei den Speichern, die übrigens zu wenig erwähnt werden, fehlen gänzlich EEPROMs.
- Ausgehend von der falschen Angabe zur Anwendungsbreite von Bauelementen werden der Computersektor überbetont und die anderen Anwender unterbewertet.

Dr. Rolf Wätzig

DUMP-Format

Für kürzere Programme ist der Assembler-Quelltext bzw. vielleicht besser noch die Übersetzungsliste wohl die optimale Darstellungsart. Längere Programme lassen sich vom HEX-DUMP aber wesentlich schneller übernehmen und benötigen viel weniger Platz. Um Fehler bei der Übermittlung auszuschließen, schlage ich vor, daß Autoren Ihnen die Programme direkt per Kassette zusenden.
Nach einer einheitlichen Formatierung könnte dann mit einem hochwertigen Drucker, den nun mal nicht jeder besitzt, die Druckvorlage hergestellt werden. Vielleicht geht es auch direkt vom Computer auf die Setzmaschine?
Sinngemäß müßte sich dies auch mit Assemblertext verwirklichen lassen. Leider hat sich bei uns für Kleincomputer kein einheitliches Aufzeichnungsformat durchgesetzt, so daß solche Vorhaben, aber auch der allgemeine Programmaustausch, erschwert sind. Vielleicht könnte durch Sie für diese Zwecke einmal eine Art Standardkassettenroutine veröffentlicht werden, die sich auch an U880-Rechner mit minimaler Hardware anpassen lassen sollte (evtl. Z1013-Routine).
Der Beitrag U880-Editor von Andreas Bogatz war zwar sehr interessant, hatte aber, wie leider sehr viele in unseren Fachzeitschriften, nur informativen Charakter. Das Interesse gerade an diesem Programm dürfte aber so groß sein, daß ein entsprechender HEX-DUMP mit Anpassungshinweisen für andere Rechner sicherlich von vielen mit Dank aufgenommen werden würde.

Helmar Thiere

Leider verfügen wir noch nicht über die Hardware und die entsprechende Software, um Manuskripte auf Kassette oder Diskette entgegennehmen und bearbeiten zu können. Wenn diese Voraussetzungen gegeben sind, ist als nächster Schritt natürlich denkbar, daß die bearbeiteten Beiträge direkt vom Redaktionscomputer per Diskette an die Lichtsatzanlage gehen. Das dürfte für die Redaktion MP aber noch etwas „Zukunftsmusik" sein.

Achten Sie also bitte weiterhin darauf, daß Programmausdrucke reprofähig sind. Steht kein Drucker zur Verfügung, tut's in Ausnahmefällen auch mal die Schreibmaschine. Sehr wichtig sind auf jeden Fall die Verwendung von weißem Papier und schwarzem Farbband. Und in diesem Zusammenhang noch eine Bitte an alle Autoren: Reichen Sie uns keine handschriftlichen Manuskripte ein. Derartige Manuskripte können wir nicht bearbeiten, Sie erhalten sie in Zukunft postwendend zurück. Zu ihrer zweiten Frage können wir Ihnen mitteilen, daß die Kleincomputer KC 85/1, /2, /3, und KC 87 ein einheitliches Kassetteninterface besitzen. Dieses Interface kann mit relativ geringem Hardwareaufwand an andere U880-Systeme angepaßt werden. Die Unterlagen werden vom VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen, Eisenacher Straße 40, 5700 zur Nachnutzung angeboten.

Arbeitskreis FORTH

Im Rahmen der Interessengemeinschaft FORTH (KDT Suhl) wurde ein Arbeitskreis FORTH-Standards/Empfehlungen gegründet. Aufgabe dieses Arbeitskreises soll es sein, FORTH-Lösungen zu Standard-Problemen der Programmierung (z. B. Arithmetik, Datenstrukturen, Listenverarbeitung, Entwicklungshilfsmittel, Nutzerkommunikation, Grafik) abzustimmen und zu publizieren, um sie einer breiten Anwendung zugänglich zu machen und so die Effektivität und die Austauschbarkeit von FORTH-Lösungen zu verbessern. Die Vorschläge müssen mit dem Standard FORTH-83 verträglich sein und sollten in Form von Glossaries, ergänzt durch eine Begründung sowie eine Diskussion der Vor- und Nachteile, veröffentlicht werden.
ZKI der AdW der DDR, Bereich Bildanalyse, Dr. Westendorf, Kurstraße 33, Berlin, 1086

Dr. Christian-M. Westendorf

MP im Jahr 1988

- Bereits in diesem Heft beginnen wir im MP-Kurs mit einer Reihe zum Mikroprozessorsystem K 1810 WM 86 (8086), das wegen der Verwendung im A 7150 und ESER-PC (EC 1834) für viele von besonderem Interesse sein dürfte.
- In einer ab MP 3/88 beginnenden Artikelserie werden dann die Mikroprozessoren der Reihe 80×86 ausführlich beschrieben.
- Neben anderen Vorhaben wird für MP-Kurs bereits eine Reihe zur künstlichen Intelligenz vorbereitet.
- Ein Heft wird der Problematik fehlertolerante Mikrorechnersysteme gewidmet sein, ein anderes wird die ausführliche Beschreibung des Floppy-Disk-Controllers mit Applikationshinweisen beinhalten.
- Wir werden uns weiterhin bemühen, von Tagungen und Konferenzen unseres Fachgebietes die interessantesten Beiträge für Sie zu veröffentlichen; beispielsweise von der Tagung Computer- und Mikroprozessortechnik und von der INFO'88.
- Wir rufen aber auch Sie auf, uns mit dem Angebot von noch mehr Einführungs- und Übersichtsartikeln sowie mit Beiträgen, die weniger nach dem Motto „**Was** habe ich gemacht?" als vielmehr im Stil von „**Wie** wird's gemacht?" verfaßt sind, zu unterstützen.

Natürlich nehmen wir auch weiterhin gern Ihre Hardware- und Software-Tips & Tricks zu den in der DDR verfügbaren Computern entgegen.
Mit dieser Zielstellung werden wir die Zeitschrift MP noch praxiswirksamer und interessanter gestalten.

Ihre Redaktion MP

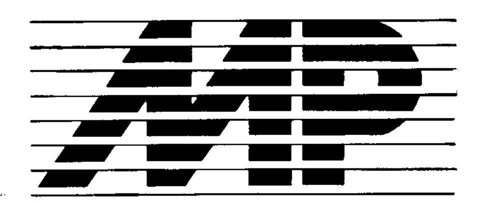
In eigener Sache

Ab sofort suchen wir für die Stelle eines Redakteurs unserer Zeitschrift MP eine(n) geeignete(n) Mitarbeiter(in) mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik (Hard- und Software).

Zu den Aufgaben gehören u. a.:
- Betreuen des Sachgebietes Computertechnik in der Zeitschrift
- Gewinnen und redaktionelles Bearbeiten von Manuskripten
- Besuchen und Auswerten von Fachtagungen, -messen und -ausstellungen
- Zusammenarbeit mit Gutachtern und ggf. selbständiges Testen von Programmen, die der Redaktion zur Veröffentlichung eingereicht wurden
- Bearbeiten bzw. Beantworten von Leseranfragen.

Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, rufen Sie uns unter Tel. 2 87 02 03 oder 2 87 03 71 an oder schreiben Sie an:
VEB Verlag Technik, Redaktion MP, Oranienburger Str. 13/14, Berlin, 1020

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232-2892



Mikroprozessortechnik, Heft 2 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR-1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 2 87 00, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Ingo Paszkowsky, Verantwortlicher Redakteur (Tel.: 2870203); Hans Weiß, Redakteur (Tel.: 2870371); Sekretariat Tel.: 2870381

Gestaltung Christina Kaminski (Tel.: 2870288)

Titelbild Tatjana Stephanowitz

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

Redaktionsschluß: 10. Dezember 1987

AN (EDV) 49837

Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tísku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Előfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Há Nội; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR-7010, und Leipzig Book Service, Talstraße 29, DDR-7010 Leipzig



Seite 35

Neues zu FORTH
Auf die Veröffentlichungen in MP 6/87 zur Programmiersprache FORTH erreichten uns zahlreiche Lesermeinungen mit der Bitte, FORTH doch etwas ausführlicher vorzustellen. Wir hoffen, mit den drei vorliegenden Beiträgen den Wünschen vieler Leser gerecht zu werden.

MP

16-Bit-μP K 1810 WM 86
Bereits angekündigt hatten wir, daß in diesem Heft unter der Rubrik Kurs mit einer Reihe zum Mikroprozessorsystem K 1810 WM 86 (8086) begonnen wird.
In Anbetracht der großen Bedeutung, die dieses Mikroprozessorsystem für den Computereinsatz in der Volkswirtschaft der DDR hat, und um unseren Lesern möglichst schnell umfassende Informationen zu diesem μP-System zu übermitteln, werden die einzelnen Folgen einen Umfang von 8 Druckseiten haben.
Alle anderen Reihen werden – wie bisher üblich – weiterhin mit einem Umfang von 4 Seiten gedruckt.

MP



Seite 62

Inhalt

Softwareleistungen auf Honorarbasis

Mit dem beschleunigten Einsatz von CAD/CAM-Technik und der rasch zunehmenden Zahl von Büro- und Personalcomputern in allen Bereichen der DDR-Volkswirtschaft wächst der Bedarf an Programmen für diese Schlüsseltechnologie stark an. Eine „Anordnung zur Durchsetzung von Ordnung und Sicherheit bei der Durchführung von Softwareleistungen in nebenberuflicher Honorartätigkeit – Honoraranordnung Softwareleistungen" zielt darauf, die Initiativen befähigter Werktätiger auf diesem Gebiet zu fördern und zusätzlich zu bestehenden Kapazitäten weitere Möglichkeiten zur Erarbeitung von Programmen für die Computertechnik zu erschließen.
Die Anordnung regelt einheitlich für alle Kombinate, volkseigenen Betriebe, Staatlichen Organe und Einrichtungen, sozialistischen Genossenschaften sowie gesellschaftlichen Organisationen, in welchen Fällen die Erarbeitung von Software außerhalb der ständigen Aufgaben von Werktätigen vereinbart werden kann. Sie trifft Festlegungen zu Honorarverträgen, insbesondere zur Gewährleistung einer hohen Qualität der Software und zur Höhe der Vergütung, und legt die Pflichten der Betriebe zur Einhaltung von Ordnung und Sicherheit auf diesem Gebiet fest. Dazu gehört auch, die Zentrale Informationsbank Software im VEB Datenverarbeitungszentrum Dresden regelmäßig über abgeschlossene Softwareleistungen zu informieren, um die Programme breit nachnutzen zu können und Doppelentwicklungen zu vermeiden.
Veröffentlicht ist die Anordnung im Gesetzblatt Teil I Nr. 28 vom 16. November 1987. Sie ist seit 1. Dezember 1987 in Kraft.

ADN

mikroelektronik

Messen, Steuern, Regeln – mit elektronischen Bauelementen aus dem VEB Kombinat Mikroelektronik

Steigende Effizienz und Präzision sind sowohl beim Hersteller als auch beim Anwender von Meß-, Steuerungs- und Regelungstechnik heute mehr denn je gefragt.
Es ist daher nicht verwunderlich, daß gerade dieser Bereich eine Domäne der Mikroelektronik-Applikation wurde und weiter bleibt, sichert doch gerade die hochentwickelte Mikroelektronik Effizienz und Präzision.
Mit seinem großen und leistungsfähigen Sortiment elektronischer Bauelemente ist deshalb der VEB Kombinat Mikroelektronik der richtige Partner.
Dies wird am breiten Spektrum der zur Leipziger Frühjahrsmesse 1988 in der Halle 15 vorgestellten Bauelemente und anhand zahlreicher Applikationsbeispiele deutlich.
Für nahezu alle Aufgaben stehen die richtigen Bauelemente der Mikro-, Opto- und Leistungselektronik bereit. Das Produktionsprogramm des Kombinates enthält Mikroprozessoren mit Verarbeitungsbreiten bis 16 Bit, Einchipmikrorechner, hochintegrierte Speicherschaltkreise, Zähler, Ein- und Ausgabe-IC, AD/DA-Wandler, Operationsverstärker, Analog- und Digitalprozessoren, Ansteuerschaltkreise, Sensoren und Koppler, Fotodioden, Infrarot- und Lichtemitterdioden, Flüssigkristallanzeigen, Lichtschachtbauelemente, Si-Transistoren und Leistungstransistoren, Si-Dioden und Leistungsdioden, Gleichrichterdioden und Freiflächengleichrichter.
Nahezu 1500 Grundtypen aktiver elektronischer Bauelemente umfaßt das Lieferangebot.
Digitale Meßtechnik (Multimeter, Handmultimeter, Zähler u. a.) mit einem breiten Einsatzbereich ergänzen die Erzeugnispalette. Als kompetenter und leistungsstarker Partner in Sachen Zeitmeßtechnik präsentiert sich der VEB Kombinat Mikroelektronik als Hersteller hochpräziser quarzgesteuerter sowie mechanischer Industrie-Schaltuhren für viele Anwendungen, Marinechronometern und einer großen Uhrenkollektion, die in Funktionalität und Design überzeugen.

VEB Kombinat Mikroelektronik

Mobilar für Bildschirmarbeitsplätze

Auf der LHM '87 stellte der VEB Elektro- und Metallgeräte Ilmenau seine Bildschirmarbeitsplätze mit Stahlprofilgestell und Arbeitsplatten aus sprelacartbeschichteter Möbelspanplatte vor. Es gibt die Arbeitsplätze in den Abmessungen (in mm) 1600 × 900 × 720 und 1200 × 750 × 720. Die Rollschränke (Rollkorpus) werden auch in zwei Ausführungen hergestellt, nämlich 420 × 790 × 640 mm³ und 420 × 605 × 640 mm³.

MP

Fotos: Grunder (2)



Das Kombinat VEB Elektro-Apparate-Werke auf der LFM '88

Wenn am 13. März die Leipziger Frühjahrsmesse 1988 ihre Pforten öffnet, wird ihren Besuchern in der Halle 15 auf dem Gelände der Technischen Messe eine Exponatenschau elektronischer Automatisierungstechnik präsentiert, die Problemlösungen für jeden Wirtschaftszweig zeigt. Seit Jahren ist das Kombinat VEB Elektro-Apparate-Werke größter DDR-Produzent dieser für die Rationalisierung und Automatisierung technologischer Prozesse erforderlichen meßsteuer- und regelungstechnischen Geräte. Das K EAW stellt sich in der Halle 15 wieder mit einem ausgewählten Spektrum seines etwa 50 000 Erzeugnistypen umfassenden Produktions- und Leistungsangebotes vor. Bei seiner Präsentation legt das Kombinat Wert auf die Darstellung der Exponate der Haupterzeugnislinien EAW-electronic, EAW-sensoric und EAW-actoric sowie ihr Zusammenwirken in rationellen und effektiven Problemlösungen. Im Vordergrund stehen dabei neue Exponate der EAW-electronic in anschaulich dargestellten, bereits realisierten industriellen Problemlösungen. So wird auf dem Gebiet der Steuerungstechnik das System EAW-electronic S 2000 für Transport- und Lagerprozesse in Aktion zu sehen sein, das sich beispielsweise bereits in einem Hochregalnormteillager des Maschinenbauhandels Berlin bewährt. Die Einsatzmöglichkeiten des kompakten Steuerungs- und Regelungssystems EAW-electronic S 2000 demonstrieren weitere Problemlösungen wie die „Automatisierung einer Fermentierungsanlage" als Einsatzfall in der Biotechnologie oder die Regelung der Plasmabeschichtungsanlage TINA 900-1.1 H aus dem Kombinat Carl Zeiss JENA.
Einen weiteren Schwerpunkt bildet ein multi-user-Arbeitsplatz zur Softwareerarbeitung mit dem Programmier- und Entwicklungssystem EAW-electronic P 8000. Gezeigt wird auch der automatisierte Schiffsmaschinenbetrieb mit Teilkomplexen des Systems EAW-electronic E 8100, das sich bereits mit einer Maschinenüberwachungsanlage, Manöverregistriereinrichtung und dem Hauptmaschinensicherheitssystem an Bord der 20 000-Tonnen-Vollcontainerschiffe der DDR-Handelsflotte „Ernst Thälmann" und „Wilhelm Pieck" bewährt.

Ungekapselte Chips auf Platinen

In den USA ist jetzt die Montage ungekapselter IC-Chips auf größeren Substraten, bis hin zu Printplatten kleinerer Abmessungen, Gegenstand von Fachdiskussionen geworden. Ausgangspunkt sind die derzeit populär werdenden „Smart Cards". Das sind in erster Linie Kreditkarten mit eingebauten Speicher- und Prozessor-ICs. Sie befinden sich bekanntlich auch in Europa schon in einer frühen Erprobungsphase. Im Zuge des neuen Trends bei der „COB"-Technik (Chips On Board) werden die Keramiksubstrate soweit vergrößert, daß sie nach gegenwärtigem Stand zwanzig und mehr ungekapselte ICs aufnehmen können. Gegenüber den üblichen gekapselten Bauelementen sollen sich dabei Kostenersparnisse von 80 bis 90 Prozent ergeben können. Ferner kann bei der Verdrahtung die gut beherrschte Dickschicht-Technik verwendet werden.
Noch nicht gelöste Probleme gibt es dadurch, daß das bisherige einfache Einlöten der Kapseln durch IC-Bonding-Technik in großem Maßstab ersetzt werden muß. Die bisher relativ einfache automatische Platinen-Bestückung muß damit durch komplizierte Verfahren ersetzt werden. Eine nachträgliche Reparatur fertig bestückter Boards, die defekte Chips enthalten, ist praktisch ausgeschlossen. Ein weiteres, nur erst halbwegs gelöstes Problem besteht darin, daß die montierten Chips nachträglich auf dem Substrat mit einer kapselähnlichen Schutzhülle aus Epoxid oder Silikon versehen werden müssen.
Auch die dafür erforderlichen neuen Verfahren sind bis jetzt nur erst unzureichend entwickelt worden. Schließlich erfordert die Montage der ungekapselten Chips neue Maßnahmen zur Ableitung der Verlustwärme.

aus der Elektronikschau 8/87

FORTH – eine moderne Softwarephilosophie

Thomas Höhenleitner, Berlin

1. Einführung

Mit dieser Veröffentlichung soll ein Eindruck von den Möglichkeiten des Softwarewerkzeuges FORTH vermittelt werden. Die Hinwendung zu neuen leistungsfähigen Programmierumgebungen fällt dem Assemblerprogrammierer nicht immer leicht. Einerseits ist die Vielfalt der angebotenen Werkzeuge recht groß, andererseits ist die Einarbeitungszeit nicht unerheblich und in Anbetracht zwingender Terminstellungen schwer einzuplanen. Dazu kommt eine gewisse Unsicherheit bei der Wahl des weiteren Weges, denn seriöse vergleichende Untersuchungen zu den einzelnen Programmiersprachen sind aufwendig und fehlen im allgemeinen. Wenn beim Leser Interesse und Bereitschaft geweckt werden, FORTH in den Variantenvergleich der Mittel für die Lösung eigener Aufgaben einzubeziehen, ist das Ziel dieser Arbeit erreicht. Die hohe Portabilität von FORTH-Systemen, ihre Modularität und die damit verbundene Effizienz bei der Programmierung, die Kompaktheit und die Leistungsfähigkeit dieser Sprache führen zu einer neuen Qualität in der Softwareentwicklung. Auch der Hardwareentwickler wird sich dieses Werkzeuges zur Bearbeitung seiner Probleme (Testroutinen, Hilfsprogramme, Schnittstellentreiber) bedienen wollen.
FORTH wurde Ender der 60er Jahre in den USA erfunden. Der Schöpfer Charles Moore benötigte eine effizient zu programmierende und schnell laufende Sprache für seine Steuerungsaufgaben. Das Konzept war so überzeugend, daß es seither durch eine Interessengemeinschaft, die Forth Interest Group, gefördert, verbreitet und weiterentwickelt wird. Der erste Standardisierungsversuch wurde figFORTH genannt und hat inzwischen viele Anwendungen gefunden. 1979 folgte FORTH79, erfreute sich jedoch nicht so breiter Akzeptanz. Gewachsen aus den Erfahrungen bei der Arbeit mit FORTH, wurde 1983 der Standard FORTH83 definiert, der sich zunehmend verbreitet, da er eine ganze Reihe von Verbesserungen gegenüber figFORTH und auch FORTH79 enthält. Bei den angeführten Beispielen wird bei Bedarf auf die Unterschiede zwischen figFORTH und FORTH83 hingewiesen.

1.1. Die FORTH-Maschine

Die Abarbeitung der FORTH-Instruktionen übernimmt eine Maschine, deren Befehlssatz aus den FORTH-Worten besteht. Dabei ist es für die Funktion ohne Belang, ob die Maschinenregister und ihr Zusammenspiel (innerer Interpreter) direkt in Hardware oder softwaregestützt durch einen Mikroprozessor realisiert sind. Der FORTH-Prozessor ist im wesentlichen eine Stackmaschine, die auf die Abarbeitung des FORTH-Codes abgestimmt ist. Die wichtigsten Register des FORTH-Prozessors seien kurz vorgestellt:

IP *Instruction Pointer*
zeigt in die Wortadreßkette (s. 3.5.1.) des gerade in Abarbeitung befindlichen Wortes.

W *Wortadreßregister*
hält die gerade aktuelle Wortadresse, das ist der Inhalt der von **IP** indizierten Zelle.

PC *Programm-Counter*
Befehlszähler

RP *Returnstack Pointer*
zeigt auf TOR (Top **of R**eturnstack).
Hier befinden sich Rückkehradressen bzw. Schleifenparameter der rufenden Ebenen.

DP *Datenstack Pointer*
zeigt auf TOS (**T**op **of S**tack).
Das ist der Puffer der gerade zu verarbeitenden Daten. Der Parameterstack übernimmt gleichzeitig die Arbeitsregisterfunktion.

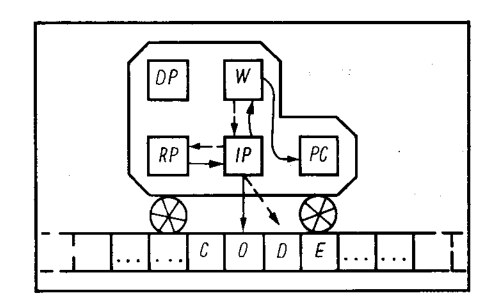


Bild 1 Eine FORTH-Maschine

Die funktionalen Zusammenhänge der Register der FORTH-Maschine (siehe hierzu Bild 1) werden im Abschnitt 3.5.5. skizziert. Ein genaues Verständnis an dieser Stelle ermöglicht dem FORTH-Anwender den Einsatz spezieller Programmiertechniken, mit denen sich u. a. auch besonders zeitkritische Probleme beherrschen lassen.

1.2. FORTH in Hardware

Der Trend zu immer größeren Verarbeitungsbreiten hat technologische und wohl auch programmtechnische Grenzen. Jüngste Entwicklungen auf dem Gebiet der Prozessortechnik zeigen, daß mit intelligenteren Architekturen Rechengeschwindigkeiten erreicht werden, die weit über den Werten üblicher Prozessoren liegen. RISC-Prozessoren (**Re**duced **I**struction **S**et **C**omputing) seien kurz genannt. Eine in /2/ und /3/ beschriebene Hardwarerealisierung eines FORTH-Prozessors in CMOS-Gate-Array-Technik zeigt eine weitere interessante Entwicklungsrichtung. Die 25fache Abarbeitungsgeschwindigkeit (!) dieser FORTH-Maschine gegenüber einem C-Programm auf einem 16-Bit-Prozessor (INTEL 8088), wie sie am Beispiel des Siebs des Erasthotenes (Primzahlermittlung) festgestellt wurde (/2/), gibt eine Vorstellung von der Leistungsfähigkeit einer solcherart strukturierten Hardware, deren Hersteller für die mit 4,5 MHz getaktete CPU 6 MIPS (**Millionen** Instruktionen **per S**ekunde) angibt. Weitere Ausführungen zu Hardware-Realisierungen von FORTH findet man u. a. in /4/.

1.3. FORTH auf einem Hostrechner

FORTH-Systeme wurden für alle gängigen Prozessoren und Rechner implementiert. Der Assemblercode umfaßt i. a. nicht mehr als 2 KByte und ist allgemein zugänglich (z. B./5/, /13/). Etwa 80% des FORTH-Kernes bestehen aus FORTH-Secondaries, d. h. Worten, die ihrerseits aus FORTH-Worten definiert wurden (vgl. 2.1.1.). Die Anpaßarbeiten an einen beliebigen Rechner, dessen Prozessor dem einer vorhandenen FORTH-Version entspricht, beschränken sich auf die Systemschnittstellen bei einigen wenigen FORTH-Worten. Damit wird klar, wie die hohe Portabilität von FORTH zustande kommt. Oft gibt es für einen Rechner verschiedene Implementierungen, welche sich in ihrem Auf- und Ausbau erheblich unterscheiden können.
Für die Verwendung der prozessorinternen Register gibt es Empfehlungen, die jedoch nicht unbedingt eingehalten werden müssen. Wie gut sich verschiedene Mikroprozessoren als Stackmaschine einsetzen lassen, hängt von der speziellen Registersteuerung ab /1/, /6/, /7/. Hier findet man auch den Schlüssel für Programmlaufzeituntersuchungen, wenn derartige Betrachtungen bei zeitkritischen Applikationen erforderlich werden sollten.

2. Die Programmiersprache FORTH

2.1. Offenes System

FORTH ist erweiterbar – nicht nur im Sinne von zusätzlichen Funktionen schlechthin. Programmieren in FORTH heißt FORTH erweitern, bis letztendlich der Programmname Bestandteil des FORTH-Vokabulars geworden ist. Der unterschiedliche Begriffsinhalt von Worten wie Operand, Befehl, Routine, Konstante, Variable, Feld, Programm, Vokabular verschmilzt zu einer eine neue Denkweise provozierenden Einheitlichkeit. Man kann sich FORTH als ein Baukastensystem vorstellen, aus dessen Elementen beliebige neue Elemente erzeugt werden. Alle Elemente, vorgegebene wie neu erzeugte, sind ständig verfügbar; sowohl für den interaktiven Dialog als auch für die Generierung neuer Elemente. Das erzeugte Programm stellt letztendlich auch nur ein weiteres Element neben beliebigen anderen dar. In dieser Modularität offenbart sich ein entscheidender Vorteil für die Programmentwicklung. Probleme werden in ihre Bestandteile zerlegt und damit besser verstanden. Jedes Element kann für sich analysiert und bearbeitet werden, ist einzeln testbar, wartbar und jederzeit wiederverwendbar. Unterprogrammaufrufe etwa in Form von „Call name" sind unbekannt. Die FORTH-Worte werden wie in einer natürlichen Sprache aneinandergefügt.

2.1.2. Definitionswörter

FORTH bietet verschiedene Definitionswörter, mit deren Hilfe das System jederzeit um neue Befehle bereichert werden kann. Die gebräuchlichste Erweiterungsform ist die Colon-Definition, deren Anwendung bei der Definition eines Hilfswortes .S zur Anzeige der oberen 3 Stackzellen demonstriert werden soll:

: .S 3 0 DO ROT DUP . LOOP ;⟨RET⟩**OK**

Eingaben werden generell jeweils durch mindestens ein Leerzeichen getrennt. ⟨RET⟩ bedeutet RETURN-Taste drücken. Terminalausgaben sind unterstrichen.

: eröffnet die Definition
.S ist ein frei wählbarer Name (dot Stack)

3 0 DO eröffnet DO ... LOOP Schleife für 3 Durchläufe
ROT ROTiert die oberen 3 Stackelemente
DUP DUPliziert das obere Stackelement
. dot transportiert oberes Stackelement zum Terminal
LOOP Ende der Schleife
; Ende der Definition

.S kann sofort getestet werden ...

7 2049 12⟨RET⟩ **OK** (3 Werte auf Stack)
.S⟨RET⟩**7 2049 12 OK**
138 .S⟨RET⟩ **2049 12 138 OK**

... und in alle weiteren Definitionen eingebaut werden. Wird die Stackanzeige in mehreren Zahlenbasen gewünscht, kann man weiter definieren (Einbau des eben kreierten Wortes .S):

```
: DS  BASE @                     (aktuelle Zahlenbasis)
      >R                                      (Retten)
      CR DECIMAL   ."dezimal           ".S
      CR HEX       ."hexadezimal       ".S
      CR 2 BASE !  ."binaer            ".S
      R>           (Restaurieren der)
      BASE ! ;     (Zahlenbasis und
                   Ende)
```

Test des neuen Wortes:

DS⟨RET⟩

dezimal	**2049 12 138**
hexadezimal	**801 C 8A**
binaer	**100000000001 1100**
	10001010 OK

Es gibt komfortablere Stackdumps, darum geht es hier jedoch nicht. Diesen mit einer Doppelpunktdefinition generierten Worttyp bezeichnet man als Secondary.
Ein weiteres Definitionswort ist z. B. **VARIABLE**. Damit kann man beliebige RAM-Zellen symbolisch unter einem Namen ansprechen.

VARIABLE MINUTEN ⟨RET⟩ **OK**[1]

weist dem neuen FORTH-Wort MINUTEN eine bestimmte Adresse im RAM zu. Der Aufruf 'MINUTEN' stellt diese Adresse auf dem Stack bereit und ermöglicht zusammen mit Worten wie

!	RAM-Schreiben	(wert addr→)
+!	RAM-Addieren	(wert addr→)
@	RAM-Lesen	(addr→wert)
?	RAM-Ausgeben	(addr→)

eine bequeme Handhabung dieser Variablen:

4 MINUTEN !⟨RET⟩ **OK**	(setzt die Zeit)
3 MINUTEN +!⟨RET⟩ **OK**	(erhöht die Zeit)
MINUTEN ?⟨RET⟩ **7 OK**	(zeigt die Zeit)

(Das Wort **?** findet man nicht unbedingt in allen FORTH-Implementierungen, da es nur im figFORTH standardisiert wurde. Man kann es im Bedarfsfall mit **: ? @ . ;** selbst definieren.)
FORTH bietet darüber hinaus die Möglichkeit, eigene Definitionsworte zu erzeugen. Die Verarbeitung sehr spezieller Datentypen ist damit problemlos möglich. Zum Beispiel lassen sich mit der **CREATE ... DOES**⟩-Konstruktion auf einfache Weise Definitionsworte generieren, mit denen man den erzeugten Worten bestimmte Compile- und Laufzeiteigenschaften zuordnen kann. Wie das in etwa funktioniert, soll ein Beispiel verdeutlichen:

[1] Gilt für FORTH83. **VARIABLE** in figFORTH-Versionen erwartet einen Initialisierungswert auf dem Stack, also:
0 VARIABLE MINUTEN⟨RET⟩ **OK**

Bei der Erstellung eines Simulationsprogrammes für elektronische Schaltungen sind u. a. verschiedene Operationsverstärker auf bequeme Art und Weise einzubinden. Der Aufruf der Typbezeichnung soll die Adresse der speziellen Operationsverstärkereigenschaften, z. B. Offsetspannung U_{off} und Leerlaufverstärkung v_{leer}, im vereinbarten Zahlenformat auf dem Stack hinterlegen. Für die bequeme Definition vieler solcher Typen wird das Definitionswort OV-TYP generiert (wie, führt hier zu weit), dessen Anwendung etwa in der Form

0.015 15000. OV-TYP B080
0.0005 25000. OV-TYP B087
0.006 25000. OV-TYP B176

erfolgt. Das Compilezeitverhalten dieses neuen Definitionswortes besteht in der Compilierung der jeweils auf dem Stack bereitgestellten OV-Eigenschaften in das Parameterfeld (s. 3.5.1.) der damit definierten OV-Typbezeichnungen. Sein Laufzeitverhalten bewirkt, daß der Aufruf einer solchen OV-Typbezeichnung die Adresse seines eigenen Parameterfeldes auf dem Stack hinterläßt, also einen Pointer auf die jeweiligen OV-Eigenschaften. Weiter läßt sich ein Definitionswort OPV für die einzelnen OV-Bauelementebezeichnungen der Schaltung mit den folgenden Eigenschaften deklarieren.

Compile-Zeitverhalten:

(Aktivität bei Aufruf des Definitionswortes OPV).
Auf dem Stack wird ein Pointer auf die speziellen Datenblattkennwerte des betreffenden Bauelementetyps erwartet (bereitgestellt durch den Aufruf einer definierten Typbezeichnung) und in das Parameterfeld der zu erzeugenden Worte compiliert. Die einzelnen Operationsverstärker der Schaltung werden also mit den „individuellen Eigenschaften" eines OV-TYPs versehen. Die Anwendung bei der Definition der Bauelementebezeichnungen sieht dann so aus:

B080 OPV 1N1	**B080 OPV 2N1**
B080 OPV 3N1	
B087 OPV N2	**B176 OPV N3**

Laufzeitverhalten:

(Aktivität bei Aufruf eines mittels OPV definierten Wortes).
Der Aufruf der einzelnen definierten Bauelemente könnte dann folgendes bewirken: Alle Spannungspunkte der zu simulierenden Schaltung sind z. B. als Variablen vereinbart. Mit dem Aufruf von beispielsweise N2 werden die die zugehörige Eingangsspannung haltenden Variablen N2U+ und N2U− entsprechend der im Definitionswort OPV vereinbarten Simulationsfunktion verknüpft, und das Ergebnis wird in die zugehörige Ausgangsspannungsvariable N2UA gespeichert. Dieses Laufzeitverhalten ist für alle OV-Bauelementebezeichnungen entsprechend gleich. Der Code existiert nur einmal (im Parameterfeld von OPV).
Das Wesen dieser Programmiertechnik besteht in einer Erweiterung des FORTH-Compilers (s. 3.5.4.) mit der genannten **CREATE ... DOES**⟩-Konstruktion. Die Vorteile liegen in der Vermeidung von redundantem Code und der besseren Programmlesbarkeit. Die Zusammenhänge sind vielleicht nicht sofort erfaßbar. Es geht hierbei auch nur um die An-

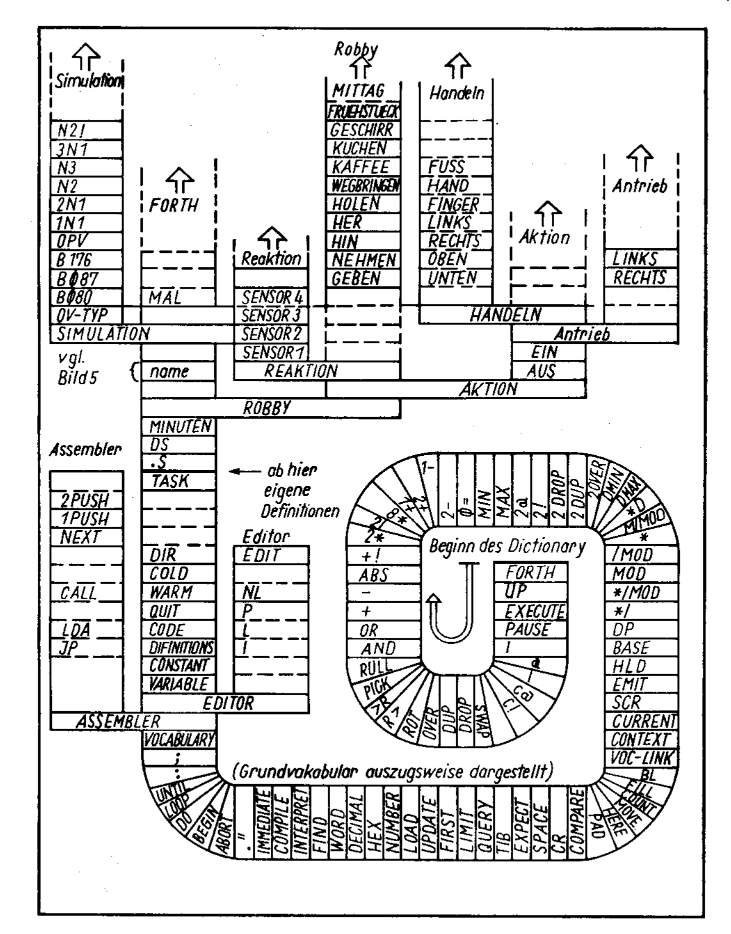


Bild 2 Baumstruktur eines FORTH-Dictionary

deutung der Vielfältigkeit der Möglichkeiten, die sich dem Programmierer bei der Softwaregestaltung mit FORTH bieten.

2.1.2. Vokabulare

Die gesamte Wortmenge eines FORTH-Systems, das Dictionary, ist in einzelne Vokabulare unterteilt und baumähnlich strukturiert (Bild 2). Ein Vokabular kann man sich als einen Zweig vorstellen. Jederzeit kann mit **VOCABULARY vocname IMMEDIATE** (s. 3.5.4.) ein weiteres, vorerst leeres Vokabular an der gerade in Erweiterung befindlichen Stelle vereinbart werden. Die Verwendung unterschiedlicher Vokabulare ermöglicht eine übersichtliche Befehlsstruktur sowie kollisionsfreie mehrfache Vergabe des gleichen Namens. Das wird z. B. schon dann interessant, wenn bei unterschiedlichen Zahlenformaten die gleichen Symbole für die entsprechenden Operatoren Verwendung finden sollen. An welcher Stelle mit der nächsten Definition zu erweitern ist, entnimmt das System der Variablen **CURRENT**, die mit der Anweisung **vocname DEFINITIONS** auf das gewünschte Vokabular gesetzt wird. Wörterbuchsuchläufe beginnen immer an der mit der Variablen **CONTEXT** spezifizierten Stelle. Diese Variable zeigt auf das zuletzt aufgerufene Vokabular.
Ein Beispiel soll die Arbeitsweise mit Vokabularen zeigen:
Einem Roboter soll der Begriff HOLEN beigebracht werden. Die Vokabulare AKTION, ANTRIEB, REAKTION gelten als vereinbart. Die Konstanten RECHTS und LINKS stellen die I/O-Adressen zur Ansteuerung zweier Motoren. EIN sendet an die auf dem Stack vorgefundene I/O-Adresse eine 1 und AUS erledigt analog das Gegenteil (hier ist u. U. bereits Maschinencode). Die Worte SENSOR1 und SENSOR2 erzeugen ein Flag auf dem Stack, welches von **UNTIL** verwertet wird. War es 0, so noch einmal **BEGIN**nen, bis die erwartete 1 kommt und die Schleife verlassen wird. Mit zwei einfachen Doppelpunktdefinitionen lassen wir nun unseren hypothetischen Roboter HIN und HER fahren.

```
: HIN AKTION ANTRIEB RECHTS EIN
      BEGIN REAKTION SENSOR1 UNTIL
      AKTION ANTRIEB RECHTS AUS ;
: HER AKTION ANTRIEB LINKS EIN
      BEGIN REAKTION SENSOR2 UNTIL
      AKTION ANTRIEB LINKS AUS ;
```

Analog hat „Robby" noch NEHMEN (d. h., er nimmt ein Teil, dessen Kennzahl er auf dem Stack erwartet) und GEBEN (er läßt alles fallen) gelernt. Mit der folgenden Zeile kann er dann alles HOLEN, was ihm irgend erreichbar ist und natürlich wieder WEGBRINGEN:

```
: HOLEN        HIN NEHMEN HER GEBEN ;
: WEGBRINGEN   HER NEHMEN HIN
               GEBEN ;
```

Mit dem Definitionswort **CONSTANT** deklariert man Kennzahlen, unter denen man z. B. Schubfachnummern vereinbaren kann:

```
7 CONSTANT KAFFEE
12 CONSTANT KUCHEN
3 CONSTANT GESCHIRR
```

Noch eine kleine Definition, und es kann FRUEHSTUECK geben:



Bild 3 Robby (Zeichnung: Jörg Olberg)

: FRUEHSTUECK KAFFEE HOLEN KUCHEN HOLEN ." Bitte zu Tisch!" ;
FRUEHSTUECK ⟨RET⟩ (Kaffee und Kuchen kommen) **Bitte zu Tisch! OK**
GESCHIRR WEGBRINGEN⟨RET⟩ (klirr) **OK**

Vergleiche hierzu die Bilder 2 und 3. Natürlich ist in so einem einfachen Fall der Einsatz so vieler Vokabulare nicht gerechtfertigt. Man denke jedoch auch an sehr umfangreiche Applikationen, wo die Möglichkeit, Namen mehrfach vergeben zu können, unschätzbare Vorteile bringen kann.

2.1.3. Assemblerprogrammierung

FORTH-Systeme enthalten i. a. einen Assembler, der die direkte Verwendung der prozessorspezifischen Mnemonik bei der Definition von Primitives ermöglicht. Primitives sind FORTH-Worte, die keine weiteren FORTH-Worte mehr aufrufen, sondern direkt Maschinencode zur Ausführung bringen. Der Assembler ist in einem Vokabular mit dem Namen **ASSEMBLER** untergebracht. Er enthält zusätzlich Strukturierungselemente wie die **IF... ELSE... THEN**[2] oder **BEGIN... WHILE... UNTIL**-Konstruktionen. Die Definition eines Primitives erfolgt mit der Anweisung **CODE name... (Mnemonics)... END-CODE**. Assemblercode kann auch zusammen mit FORTH-Worten innerhalb einer Definition generiert werden. Man formuliert: **: name... (FORTH-Worte)... ;CODE... (Mnemonics)... END-CODE.** Es lassen sich auch umfangreiche bereits vorhandene Assemblerprogramme unter einem beliebigen Namen in das System einbinden, wenn man diese in den Speicher lädt und mit einem CALL die Kontrolle an das jeweilige Programm übergibt. Die Rückkehr aus der Primitive-Ebene in das FORTH-System erfolgt mit dem Sprung nach **NEXT**, womit die Kontrolle wieder an den inneren Interpreter (3.5.5.) übergeben wird.

2.2. Datenhandling

2.2.1. Parameterübergabe

Bei Softwareprojekten ist u. a. die Frage der Parameterübergabe zwischen den einzelnen Modulen zu klären. Jede Routine muß wissen, wo die Quelldaten liegen und auf welche Art und Weise die Ergebnisparameter übergeben werden. Hierfür können Variablen oder Register vereinbart werden. Im Multitaskbetrieb sowie bei Interruptanforderung ist Zwischenspeicherung erforderlich. Daten und Adressen liegen völlig durcheinander auf einem Stack und ergeben recht unübersichtliche Systemzustände, was u. a. die Fehlersuche nicht gerade erleichtert. FORTH begegnet diesem Problem wirkungsvoll durch die Verwendung von 2 Stacks. Die getrennte Ablage von Rückkehradressen auf einem Returnstack und Daten auf einem Parameterstack (auch Datenstack oder einfach nur Stack genannt) machen das System überschaubarer. Die generelle Vereinbarung des Datenstacks als Übergabemedium ermöglicht einfachen, unsichtbaren Parametertransfer und entlastet den Programmierer von der Deklaration der Art und Weise der Datenübergabe zwischen den einzelnen Modulen. Das FORTH-Grundvokabular ist so angelegt, daß die jeweils bereitgestellten Parameter von den aufgerufenen Worten vernichtet und Ergebnisparameter für nachfolgende Worte in verarbeitungsgerechter Reihenfolge bereitgestellt werden. So nimmt z. B. das Wort **C@** (Byte-Lesebefehl) eine Adresse von der obersten Stackzeile und hinterläßt stattdessen dort im Low-Byte den gelesenen Wert (High-Byte = 0). Der Returnstack ist für die internen Rücksprungadressen und Schleifenparameter reserviert und braucht i. allg. nicht beachtet zu werden. Dieser Stack kann, unter Berücksichtigung der Strukturierungsanweisungen (**IF THEN**, **DO LOOP**,...), innerhalb eines Wortes als Zwischenspeicher Verwendung finden, aber auch zum Überspringen von Verarbeitungsebenen gezielt manipuliert werden. Man verwendet dazu die Worte
⟩R (Top of Stack → **Returnstack**)
R⟩ (**Returnstack** → Top of Stack)

2.2.2. Massenspeicher

FORTH-Dialogsysteme sind i. a. floppyorientiert. Die Massenspeicheranbindung erfolgt nach dem Prinzip des virtuellen Speichers und wird in den verschiedenen FORTH-Systemen unterschiedlich gehandhabt. figFORTH und ähnliche Systeme kennen i. allg. kein Dateihandling und teilen die Disketten in gleichgroße Blöcke entsprechend dem vereinbarten Diskettenformat auf. Der Zugriff auf die Daten ohne Verwendung einer Directory direkt über Spur- und Sektornummer ermöglicht einfache Disktreiberroutinen und extrem kurze Zugriffszeiten. Andere Implementierungen sind an ein Betriebssystem gebunden und arbeiten mit Dateien (z. B. /12/). Speicherbereiche lassen sich so einfacher zwischen unterschiedlichen Systemen transferieren, die Übersicht über den Disketteninhalt ist besser, und es brauchen keine besonderen Disketten eingerichtet zu werden. Innerhalb der gerade vereinbarten Datei verhält sich das System genauso wie eine Implementierung mit Massenspeicheranbindung ohne Dateistrukturierung. Betriebssystemspezifische Befehle lassen sich über einen System-Call problemlos als FORTH-Worte einbinden, so daß z. B. bei Diskettenwechsel aus FORTH heraus mit DIR die Directory gelistet werden kann.

---

[2] Oft ist anstelle von **THEN** (FORTH83) die Bezeichnung **ENDIF** anzutreffen (figFORTH).

[3] Kleinste Stackeinheit, umfaßt 2 Byte

```
B:ROOT.SCR  Screen 26
 Ø ( ASCII-DUMP  .ASCII  .ADR                              hoe 2Ø.Ø7.87 )
 1 : .ADR   BASE @ >R HEX Ø 4 D.R R> BASE ! ;  ( gibt Adresse aus)
 2
 3 : .ASCII                 ( adr --> ) ( gibt ASCII oder SPACE aus)
 4    C@ DUP 32 < OVER 127 > OR
 5    IF DROP 32 ENDIF EMIT ;
 6
 7 : ASCII-DUMP  ( start laenge --> ) ( gibt ASCII oder SPACE aus)
 8          BASE @ >R HEX OVER + OVER        ( start ende start --)
 9    DO    I OVER -  64 MOD Ø=             ( 64 Zeichen pro Zeile)
1Ø          IF    CR 8 SPACES                          ( neue Zeile)
11                I .ADR  SPACE                  ( Adresse ausgeben)
12          ENDIF
13          I .ASCII                        ( Drucke ASCII oder SPACE)
14    LOOP
15          DROP R> BASE ! ;                              ;S

 ok
```

Bild 4 Screenbeispiel

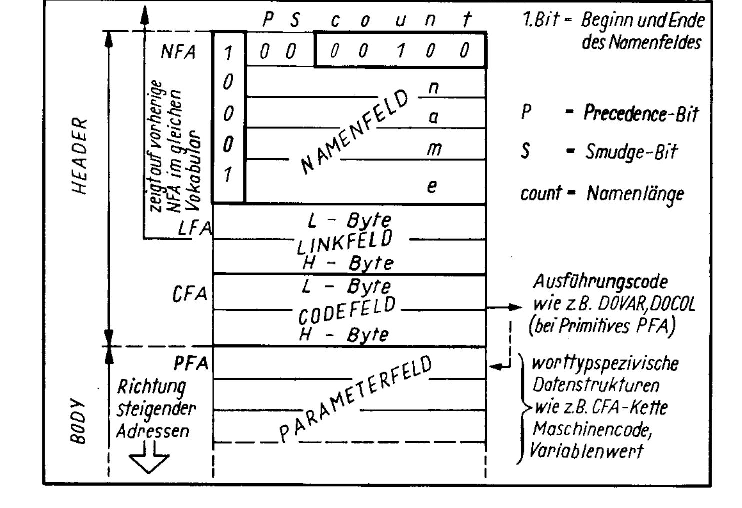


Bild 5 Lexikoneintrag

2.2.3. Screens

Ein Screen ist ein 1 KByte großer Speicherbereich und dient der Aufnahme von Quelltext oder Daten. Im RAM des Systems ist ein Bereich vereinbart (Block-Buffer-Area), der einen oder mehrere Screens aufnehmen kann. Die Screens werden vom Massenspeicher (i. allg. Diskette) geladen und dort z. B. mit einem Editor manipuliert oder entsprechend anderen Anweisungen verarbeitet. Der Anwender braucht sich um die Verwaltung der Buffer-Area nicht zu kümmern. FORTH kennzeichnet mit **UPDATE** einen Screenpuffer bei Veränderung desselben, wodurch der Screen automatisch zurückgeschrieben wird, wenn der Puffer für einen anderen Screen benötigt wird. Die Größe der Block-Buffer-Area kann aus den Variablen **FIRST** (Beginn) und **LIMIT** (Ende) errechnet werden. Ein Screen ist i. allg. in 16 Zeilen zu 64 Zeichen unterteilt. Zeile 0 sollte als Kommentarzeile reserviert werden. Mit z. B. **15 45 INDEX'** (listet die Zeilen 0 der Screens 15 bis 45 auf) ist dann eine schnelle Inhaltsübersicht über den ausgewählten Bereich möglich.

2.2.4. Editieren des Quelltextes

Üblicherweise wird der FORTH-Quelltext in Screens (Bild 4) notiert. Das geschieht unter Verwendung eines Screen-Editors, i. allg. selbst in FORTH geschrieben. Einen Line-Editor (zeilenweises Editieren) bietet fast jedes System; Full-Screen-Editoren (Bildschirm-Editoren) ermöglichen wesentlich komfortableres Arbeiten, sind aber nicht immer problemlos auf anderen Rechnern lauffähig. Die Anpassung bereitet jedoch aufgrund der Modularität von FORTH keine Schwierigkeiten. Die Verwendung von Screens zur Aufnahme von Quelltext entspricht der FORTH-Philosophie des modularen Aufbaus aller Bestandteile (also auch des Quelltextes), findet jedoch nicht uneingeschränkte Zustimmung. Prinzipiell ist es dem System egal, aus welcher Quelle (Tastatur, Diskette, serieller Datenkanal) der Text-Eingabe-Strom kommt. Einzig die FORTH-Konventionen (Spaces zwischen den Worten, keine unbekannten Zeichenfolgen, also Steuerzeichen) müssen eingehalten sein. Der Programmierer kann seinen hauseigenen Lieblingseditor weiterverwenden und muß lediglich seinen Quelltext im nachhinein entsprechend aufbereiten. In /8/ wird das am Beispiel eines allgemein bekannten Textverarbeitungsprogramms demonstriert, indem ein Softwarefilter für die entsprechenden Steuerzeichen installiert wird.

2.2.5. Transient Program Area und vorcompilierter FORTH-Code

Bei der Arbeit mit FORTH werden bestimmte Programmteile immer wieder benötigt. So z. B. ein Editor für die Editierphase und ein Debugger für die Testphase. Diese Tools werden in fertigen Applikationen i. allg. nicht mehr benötigt. Hat man sie vor der Compilierung des erstellten Programms geladen, geht mit z. B. **FORGET ASSEMBLER** auch das Arbeitsergebnis über Bord. Die im Arbeitsergebnis enthaltenen Primitive-Definitionen lassen sich aber nur mit einem bereits geladenen Assembler compilieren... Bei kleineren Rechnerkonfigurationen kann auch der verfügbare Speicherbereich schnell ausgeschöpft sein. Will man dann z. B. Editor und Debugger abwechselnd laden, entsteht aufgrund der wiederholten Compilierung immer wieder Wartezeit, und vorbei ist es mit dem kurzen Wechsel zwischen Codierung und Test. Man kann also schnell in die mißliche Lage kommen, mehr Zeit und Aufmerksamkeit in die Verwaltung des eigenen Systems zu investieren als für die eigentliche Aufgabe. Um dem Programmierer an dieser Stelle das Leben zu erleichtern, bieten verschiedene FORTH-Systeme (z. B. /12/) eine sogenannte Transient Program Area (TPA) für unabhängig vom Wörterbuchausbau ladbare Vokabulare. Aufgrund der fest vereinbarten Anfangsadresse der TPA ist auch das Laden von vorcompiliertem FORTH-Code möglich und so ein spürbarer Zeitgewinn erreichbar.

2.3. Arithmetik

2.3.1. Notation

In-fix-Anweisungen (d. h. Operator zwischen den Operanden), wie z. B. in BASIC, müssen vom jeweiligen System vor Funktionsausführung intern in eine post-fix-Form (Operator nach den Operanden) umgesetzt werden. Aufgrund der Datenübergabe über einen Stack (2.2.1.) und der streng sequentiellen Textauswertung des Textinterpreters **INTERPRET** werden in FORTH arithmetische Operationen post-fix notiert. Diese Schreibweise ist unüblich und bedarf einiger Gewöhnung. Oft wird diese auch als UPN (umgekehrte polnische Notation) bekannte Schreibweise als Hauptargument bei einer ablehnenden Haltung gegenüber FORTH gebraucht. Wenn man die Philosophie dieser Sprache richtig verstanden hat, erkennt man in logischer Konsequenz die Notwendigkeit der post-fix-Notation. Soll auf die in-fix-Schreibweise partout nicht verzichtet werden, ist eine Erweiterung des Textinterpreters **INTERPRET** erforderlich – eine aufgrund der Zugänglichkeit von FORTH lösbare Aufgabe. Allerdings erhöht man damit den Overhead des Systems beträchtlich. Eine andere Möglichkeit zeigt dieses kleine Beispiel:

4 3 ∗ .⟨RET⟩ **12 OK** (post-fix-MAL)
: MAL 32 WORD NUMBER DROP ∗ ;⟨RET⟩ **OK**[4]
4 MAL 3 . ⟨RET⟩ **12 OK** (in-fix-MAL)

Dieses MAL ist jedoch innerhalb einer Colon-Definition nicht anwendbar und sollte mit **FORGET MAL** wieder „vergessen" werden.

2.3.2. Integerarithmetik

FORTH zielt nicht explizit auf die Offline-Verarbeitung numerischer Datenmengen, obwohl es prinzipiell auch dafür einsetzbar ist. Von Hause aus ist eine 2- und 4-Byte-Arithmetik verfügbar, die eine hohe Rechengeschwindigkeit bietet und einen Zahlenbereich von 65536 (2 Byte) bzw. über 4 Milliarden (4 Byte) überstreicht. Für sein Steuerungsproblem wünscht man sich meist kurze Reaktionszeiten. Ist z. B. für eine schnelle Pegelanpassung ein Meßwert innerhalb kürzester

---

[4] Diese Definition bezieht sich auf FORTH83. Auf figFORTH-Systemen gilt aufgrund der etwas anderen Definition von **WORD** folgende Sequenz:
: MAL 32 WORD HERE NUMBER DROP ∗ ;

Tafel 1 Scalierungsquotienten

Konstante	Approximation	Fehler
$\pi = 3{,}14159\ldots$	355/113	$< 10^{-7}$
	22/7	$< 10^{-3}$
$\sqrt{2} = 1{,}41421\ldots$	19601/13860	$< 10^{-8}$
	99/70	$< 10^{-4}$
$\sqrt{3} = 1{,}73205\ldots$	18817/10864	$< 10^{-8}$
	97/56	$< 10^{-4}$
$\sqrt{10} = 3{,}16227\ldots$	22936/7253	$< 10^{-8}$
	117/37	$< 10^{-4}$

Zeit zu logarithmieren, wäre das Auslesen einer vorberechneten Tabelle (und evtl. Interpolation) eine einfache und akzeptable Möglichkeit. In /15/ ist das anhand der Sinusfunktion gezeigt.
Der 2-Byte-Zahlenbereich ist für die Verarbeitung von Meßwerten z. B. aus einem Analog-Digital-Umsetzer ideal geeignet. Soll nun der Meßwert oder irgendein anderer Parameter mit einem beliebigen Faktor multipliziert werden (Skalierung), findet sich mit Sicherheit ein Quotient, der dem gewünschten Skalierungsfaktor bis auf einen sehr kleinen Fehler nahekommt. Tafel 1 gibt einige solche Quotienten für gängige Konstanten wieder. Weitere Werte sind z. B. in /1/, /8/ und /9/ zu finden. Mit Hilfe von */ wird aus dem Zähler und dem Stackwert ein doppelt genaues Zwischenprodukt berechnet und die Division durch den Nenner zu einer wieder einfachgenauen Zahl ausgeführt. Wie genau diese Rechnung wird, sollte man einmal mit Hilfe eines Taschenrechners nachvollziehen.

2.4. Erweiterungen

Nichtsdestotrotz wird so mancher potentielle Anwender sagen: Ich habe genügend Rechenzeit, aber keine Zeit, mir über die Realisierung von arithmetischen Funktionen und die Überschreitung von Zahlenbereichen Gedanken zu machen. Nun, es läßt sich immer ein geeignetes Gleitkommapaket finden (siehe z. B. /10/), falls es nicht ohnehin schon vorhanden ist; entweder als FORTH-Erweiterung oder, wie in /11/ demonstriert, als Einbindung einer vorhandenen Gleitkommaarithmetik. Welches Format man einsetzen wird, ergibt sich aus den Erfordernissen und vorhandenen Möglichkeiten. Für hochgenaue Berechnungen findet man z. B. in der von der WPU Rostock angebotenen com-FORTH-Version /7/, /12/ eine überlange Integerarithmetik, deren bis zu 22 Byte langes Format keine Wünsche offenlassen dürfte. Wer mit komplexen Zahlen rechnet oder mit Strings hantiert, findet ebenfalls solche speziellen Systemergänzungen. Die Erweiterungsmöglichkeiten von FORTH sind bereits jetzt recht vielfältig und werden mit der weiteren Verbreitung dieser Softwarephilosophie noch bereichert werden. Dabei sind Bestrebungen zur Standardisierung zu beobachten /16/. Bevor der Anwender sich der Mühe unterzieht, selbst Standardfunktionen zu programmieren, sollte er die verfügbare Software kennen, um unnötigen Aufwand zu vermeiden.

3. Das FORTH-System

3.1. Kompaktheit

Das System ist äußerst kompakt. Schon in 2 KByte sind Grundstrukturen realisierbar. 8 KByte reichen für ein dialogfähiges System einschließlich Editor und Assembler, und in 16 bis 32 KByte lassen sich bereits übliche Applikationen unterbringen. Die Compilierung einzelner Module in ein neues Wort benötigt nur jeweils 2 Byte Speicherplatz in dessen Parameterfeld (s. 3.5.1.). Für turn-key-Systeme[5] und Applikationen für Zielsysteme ohne Dialogfähigkeit kann man bei Bedarf alle dann überflüssigen Konstruktionen wie Wortheader (vgl. 3.5.1.) und nicht (mehr) benötigte Definitionen entfernen und eine kaum noch zu überbietende Speicherausnutzung erreichen (vgl. z. B. /14/).

[5] Das fertige Programm wird als lauffähiger Code unter dem gewünschten Namen abgespeichert und läßt den Anwender von FORTH nichts mehr spüren.

Dipl.-Ing. Thomas Höhenleitner beendete 1983 sein Studium an der Humboldt-Universität zu Berlin in der Fachrichtung Elektronik-Technologie mit der Entwicklung eines digitalen Reglers für einen Kristallzüchtungsofen. Sein bisheriges Tätigkeitsfeld umfaßt Steuerelektronik von Elektrofahrzeugen und prozeßnahe Automatisierungstechnik. Seit 1988 im Zentralinstitut für Kybernetik und Informationsprozesse der AdW tätig, liegt sein neues Aufgabengebiet im Bereich der Algorithmenentwicklung und Softwareerstellung für Systeme des maschinellen Sehens.

3.2. Geschwindigkeit

Die Abarbeitung des compilierten FORTH-Code durch die emulierte FORTH-Maschine benötigt nur etwa die vierfache Zeit /8/ gegenüber einem optimierten Assemblerprogramm. Dieser Wert variiert je nach eingesetztem Prozessortyp /1/, /6/, /7/ und gegebener Problemstellung. Die Anwendung der post-fix-Notation (vgl. 2.3.1.), das Konzept des gefädelten FORTH-Codes und die Arbeitsweise des inneren Interpreters halten den Overhead bei der Programmabarbeitung gering. Dazu kommen aufgrund der Maschinennähe und des modularen Aufbaus weitere Möglichkeiten. Angenommen, in einer Applikation werden 90% der Programmlaufzeit von 5% des Codes in Anspruch genommen (z. B. eine Such-, Sortier- oder Konvertierungsprozedur) und man möchte die Abarbeitungsgeschwindigkeit erhöhen, dann kann man, ohne die restlichen 95% des Codes irgendwie modifizieren zu müssen, diese bisher als Secondary definierte Prozedur in Assembler neu codieren und als Primitive (s. 2.1.3.) in die Applikation einbinden. Die Gesamtlaufzeit des Programms erreicht dann fast die Werte, die optimierter Maschinencode bietet.

3.3. Multitasking

Für Echtzeitprobleme, aber auch bei der Offline-Verarbeitung von Daten, ist Multitaskingfähigkeit oft unverzichtbar. FORTH bietet auch das. Alle taskspezifischen Pointer sind als **USER-VARIABLE** angelegt, also über einen Offset, der in einer Variablen gehalten wird, erreichbar. Das Umschalten zwischen den einzelnen Tasks erfolgt einfach durch Verändern dieses Offsets. Zwei Verfahren sollen kurz genannt werden. Das einfachere, z. B. in der public domain Version F83 /8/ installierte, besteht darin, daß jede einzelne Task möglichst oft einen Befehl **PAUSE** enthält, der ein Weiterschalten zur nächsten Task ermöglicht. Zu diesem Zweck enthalten in F83 verschiedene, oft auftretende Anweisungen bereits dieses Wort (z. B. die Zeichen **EMIT**tierende Prozedur). Dieses unter der Bezeichnung **robin round** bekannte Prinzip /8/ ist für die meisten Anwendungsfälle völlig ausreichend. Für zeitkritische Applikationen kann man einen sogenannten Scheduler installieren, der die Tasksteuerung entsprechend einem Zeitplan und vorgegebene Prioritäten erledigt (Timesharing). Ein für den Softwareentwickler relevantes Multitask-Anwendungsbeispiel könnte beim Debugging die Anzeige aller wichtigen Systemzustände auf einem Zweitmonitor sein.

3.4. Interruptbehandlung

Wird vom Prozessor ein Interrupt akzeptiert, so kann nach dem Retten der notwendigen Register die Serviceroutine in Maschinencode ablaufen und nach der Rekonstruktion des ursprünglichen Prozessorstatus wieder an das FORTH-System übergeben werden. Soweit der einfache Fall. Interessant wird nun die Möglichkeit, Interruptserviceroutinen in FORTH zu programmieren. Nach dem bisher Gesagten kann man die Fragestellung darauf reduzieren, wie man von der Maschinenebene aus FORTH-Worte aufrufen und wieder in die Maschinenebene zurückkehren kann (um die Interruptserviceroutine mit der Registerrestaurierung und dem prozessorspezifischen Abschlußbefehl zu beenden.). FORTH ist auch in diesem Punkt offen: Das eine Doppelpunktdefinition beendende Wort ; (IMMEDIATE-Wort, s. 3.5.4.) compiliert eine Routine zum Abschluß der durch das später aktivierte Wort hervorgerufenen Prozesse (NEXT). Um ein aus der Maschinenebene heraus aufrufbares Wort zu kreieren (welches wieder dorthin zurückkehrt), kann genau hier angesetzt werden. Es ist lediglich der Sprungbefehl nach NEXT durch ein RETURN zu ersetzen. In /8/ wird dabei der unkomplizierte Weg über die Definition eines speziellen Abschlußbefehles **;RET**, der anstelle von ; eine Definition beschließt, beschritten. Mit einem Blick in das (oft nicht vorhandene) Listing oder mit Hilfe eines Discompilers erfährt man den Aufbau von ; und kann diese wenigen Zeilen (jetzt mit RETURN anstelle von NEXT!) neu codieren. Solche Worte sollten nie von FORTH aus direkt aufgerufen werden, wenn man sein System am Leben erhalten will. Der eigentliche Aufruf in der Maschinenebene kann nach Bereitstellung der Codefeldadresse (CFA, s. 3.5.1. u. 3.5.5.) über einen CALL in das Wort **EXECUTE** hinein realisiert werden. Die hier dargestellte Möglichkeit ist nicht die einzige. Weiteres ist u. a. in /1/ zu finden.

3.5. FORTH von innen

3.5.1. Struktur eines Lexikoneintrages

Bild 5 zeigt den prinzipiellen Aufbau eines figFORTH-Wortes. Die *Namenfeldadresse* (NFA) zeigt auf den Beginn des Namenfeldes, dessen erstes Byte neben der Längenangabe (count) zwei Kennungsbits enthält. Das *SMUDGE-Bit* S ist im allgemeinen 0. Noch nicht fertig definierte Wortstrukturen enthalten S=1 und sind damit vom System nicht aufrufbar. Das *Precedence-Bit* P ist bei allen IMMEDIATE-Worten (s. 3.5.4.) gesetzt. Alle FORTH-Worte eines Vokabulars sind über die jeweilige *Linkfeldadresse* (LFA) miteinander verkettet, was für die Suchläufe des Textinterpreters erforderlich ist. Sie hält einen Pointer auf die NFA des jeweils im gleichen Vokabular zuvor definierten Wortes. Die *Codefeldadresse* (CFA), auch mit Wortadresse bezeichnet, enthält die Startadresse einer worttypspezifischen Maschinenroutine. Gelangt ein FORTH-Wort zur Ausführung, übergibt der innere Interpreter (s. 3.5.5.) die Kontrolle an diesen Maschinencode. Beispielsweise transportiert DOCON, der Aus-

führungscode des Worttyps Konstante, den Inhalt der *Parameterfeldadresse* (PFA) auf den Datenstack. Einige weitere symbolische CFA-Inhalte sind:

DOVAR: (DO VARiable) bringt die PFA auf den Stack (Worttyp Variable).

DOUSER: (DO USER-variable) bringt Summe von Inhalt der PFA und Inhalt von UP (User-Area-Pointer) zum Stack (Wortyp User-Variable).

DOCOL: (DO COLon) bringt den inneren Interpreter zur nächsttieferen Verarbeitungsebene. Dann Abarbeitung der Folge von CFAs im Parameterfeld (Worttyp Secondary).

PFA: Maschinencode befindet sich im Parameterfeld (Worttyp Primitive).

Prinzipiell kann der Maschinencode an beliebiger Stelle im RAM liegen. Er muß lediglich mit einem Sprung nach NEXT, dem Eintrittspunkt des inneren Interpreters, enden. Das Parameterfeld beginnt an der PFA. Hier ist der eigentliche Wortinhalt verborgen: Daten beliebiger Struktur, direkt ausführbarer Maschinencode oder eine CFA-Folge. Zum Beispiel:

Konstante:	Der Wert selbst.
Variable:	Der Wert selbst. (Beachte unterschiedliches Laufzeitverhalten!)
User-Variable:	Offset zum UP
Secondary:	CFA-Kette der definierten Wortfolge.
Primitive:	Maschinencode.

Der in Bild 5 gezeigte Wortaufbau ist nicht für alle FORTH-Versionen identisch. Bei F83 /8/ befindet sich die LFA vor der NFA, wodurch z. B. schnellere Dictionarysuchläufe möglich werden. Vor der LFA ist ein weiterer Zeiger VFA (*View Field-Address*) angeordnet, der auf den Quellscreen verweist, aus dem das betreffende Wort compiliert wurde. Mit **VIEW name** erreicht man den zugehörigen Quelltext. In der in /14/ beschriebenen Lösung liegt stattdessen zwischen LFA und CFA ein Pointer auf das betreffende Parameterfeld. Damit ist es möglich, eventuellen Assemblerhilfscode direkt im Codefeld unterzubringen, da dieses jetzt mehr als 2 Byte umfassen darf.

3.5.2. Memory Map

Bild 6 zeigt eine typische Memory Map. Diese Aufteilung ist stark implementations- und rechnerabhängig. Es soll lediglich eine Vorstellung vom inneren Aufbau vermittelt werden. Interessiert die Speicheraufteilung einer vorliegenden Implementation, kann man sich die Werte der einzelnen Zeiger vom System erfragen. Zum Beispiel:

RPO ?⟨RET⟩ <u>3F74 OK</u>

RPO ist eine User-Variable, welche den Kaltstartwert für den Returnstack-Pointer **RP** hält.

3.5.3. Zum Beispiel der äußere Interpreter

FORTH ist nicht nur offen nach außen, sondern auch nach innen. Es gibt keine Blackbox für den Programmierer, wie das bei anderen Programmiersprachen der Fall ist. Man kann sich überall umsehen und bei Bedarf auf einfache Art und Weise Modifikationen vornehmen. Zum Beispiel der äußere Interpreter: Nach der Systeminitialisierungsphase (Worte **COLD**, **WARM** und **ABORT**) befindet FORTH sich in einer Endlosschleife, dem äußeren Interpreter **QUIT**. Beide Stacks sind leer, und das System erwartet Zeichen (meist) von der Tastatur (**QUERY**). Sobald welche eintreffen, werden sie in den **TIB** (**T**erminal-**I**nput-**B**uffer) transportiert. Mit dem Erkennen eines ⟨RET⟩ (LFCR-Sequenz OAH ODH) wird die im **TIB** aufgebaute Stringfolge **INTERPRET**iert (Aufruf des Textinterpreters **INTERPRET**) und danach wieder mit **QUERY** auf Text gelauert (dies ist eine etwas vereinfachte Darstellung). Analog kann man die Arbeitsweise von **INTERPRET** verfolgen, wo man dann u. a. auf **EXECUTE** stößt und so zum inneren Interpreter gelangt. In gleicher Weise läßt sich das System bezüglich seiner Massenspeicheranbindung und aller anderen Komponenten beleuchten.

Bild 6 Memory Map

3.5.4. Der FORTH-Compiler

Der FORTH-Compiler wird durch verschiedene Compilerworte repräsentiert, welche als integrale Systemfunktionen im Zusammenwirken mit dem FORTH-Gesamtsystem die Quelltextcompilation vornehmen. Beispielsweise:

CREATE name
Ein Wortheader (vgl. Bild 5) wird im Dictionary aufgebaut.

, (n--)
Ein 2-Byte-Wort wird vom Stack in das Dictionary compiliert.

C, (n--)
Ein 1-Byte-Wort wird vom Stack in das Dictionary compiliert (High-Byte von n wird ignoriert).

ALLOT (n--)
Reserviert n Zellen im RAM an aktueller Erweiterungsposition im Dictionary.

IMMEDIATE
Das Precedence-Bit in der letzten Definition wird gesetzt (vgl. 3.5.1.).

Das FORTH-System unterscheidet anhand der User-Variablen **STATE**, ob es sich im compilierenden oder executierenden Mode befindet. Der Wechsel zwischen beiden Modi erfolgt mittels **]** (schaltet den Compile-Modus ein und **[** (suspendiert den Compiler, womit automatisch wieder der Execute- Mode erreicht wird). So verwendet z. B. das Definitionswort **:** u. a. die Worte **CREATE** und **]**. Die Compilerworte **[** und **SMUDGE** findet man u. a. im eine Colon-Definition beendenden Wort ;. Der Textinterpreter **INTERPRET** prüft für jedes gefundene Wort den Systemstatus (**STATE** @) und übergibt die ermittelte CFA entweder an **EXECUTE** zur Ausführung oder ruft das Wort , , welches dann die CFA ins Dictionary compiliert. Zahlen im Texteingabestrom werden im Execute-Mode zum Datenstack transportiert. Oder es erfolgt die Compilierung zusammen mit einem zahlenformattypischen Literalhandler.
Die streng sequentielle Textauswertung im FORTH-System ermöglicht u. a. das Umschalten in den Execute-Mode und wieder in den Compile-Mode mitten in einer eröffneten Colon-Definition. Diesen Umstand kann man z. B. nutzen, um Berechnungen während der Compilezeit ausführen zu lassen, um damit die Laufzeiten zu minimieren. Wurde nach der Definition eines Wortes mit **IMMEDIATE** dessen Precedence-Bit (Vorrangigkeits-Bit) gesetzt, wird es auch im Compile-Mode sofort executiert. Beispielsweise ist ; ein solches IMMEDIATE-Wort. Es gibt noch weitere Compileranweisungen wie z. B. auch die selbstdefinierten Definitionsworte (vgl. 2.1.1.). Die hier erwähnten Worte verdeutlichen jedoch die prinzipielle Funktion für ein erstes Verständnis ausreichend.

3.5.5. Der innere Interpreter

Die prinzipielle Arbeitsweise des inneren Interpreters, das ist das Zusammenspiel der Register des FORTH-Prozessors (vgl. Bild 1) bei der Interpretation des gefädelten FORTH-Codes, ist in Tafel 2 skizziert.

Tafel 2 Funktionen des inneren Interpreters

Innerer Interpreter		; Kommentar	! U880-Assembler		
NEXT:		; Entry des inneren Interpreters	! NEXT:		
		; IP hält die aktuelle physische Adresse innerhalb des	!	LD A,	(BC)
		; Parameterfeldes des gerade in Abarbeitung befind-	!	LD L,	A
		; lichen Wortes	!	INC	BC
	((IP))→(W)	; Die dort befindliche CFA wird nach W gebracht. Dies	!	LD A,	(BC)
		; ist die nächste auszuführende Anweisung.	!	LD H,	A
	(IP) := (IP) + 2	; Stelle IP auf nächste physische Adresse (Vorberei-	!	INC	BC
		; tung für NEXTes Wort).	!		
NEXT1:		; innerer Entry (z. B. für EXECUTE)	! NEXT1:		
		; W hält die zu executierende CFA.	!	LD E	(HL)
	((W))→T	; Die Codeadresse CA, d. i. die Startadresse des wort-	!	INC	HL
		; spezifischen Ausführungscode, des jetzt zu execu-	!	LD D,	(HL)
		; tierenden Wortes (z. B. DOCOL, DOVAR oder anderer	!	EX DE	,HL
		; Maschinencode), wird ausgelagert (Temporary)	!		
	(W) := (W) +2	; W wird auf PFA des nun aktuellen Wortes gesetzt.	!	; DE = CFA+1	
	T→PC	; Lade PC mit CA, damit Abarbeitung des Ausführungs--	!	JP	(HL)
		; code. Dieser endet i. a. mit einem Sprung nach NEXT	!		
		; (nächstes Wort).	!		
DOCOL:		; D. i. der allen Secondarys gemeinsame Ausführungscode	! DOCOL:		
		; (DO the COlon). Die Adresse DOCOL wird bei Aufruf	!	LD HL	,(RPP)
		; der Doppelpunktdefinition in neue CFA compiliert.	!	DEC	HL
	(RP) : = (RP) + 2	; RP auf nächste Zelle setzen.	!	DEC	HL
				LD (RPP)	,HL
	(IP)→((RP))	; Nächste physische Adresse der aktuellen Verarbei-	!	LD (HL)	,C
		; tungsebene auf Returnstack retten.	!	INC	HL
			!	LD (HL)	,B
	(W)→(IP)	; W wurde in NEXT auf PFA des aktuellen Wortes gesetzt.	!	INC	DE
		; Die dortige CFA wird dem IP übergeben und damit die	!	LD C	,E
		; nächsttiefere Verarbeitungsebene erreicht.	!	LD B	,D
	JUMP NEXT	; Und wieder von vorn (jetzt eine Ebene tiefer).	!	JP	NEXT
SEMIS:		; Diese Prozedur wird am Ende einer Doppelpunktdefini-	! SEMIS:		
		; tion mit dem Aufruf des Wortes ; in das Parameter-	!	LD HL	,(RPP)
		; feld compiliert und bewirkt die Rückkehr in die	!	LD C	,(HL)
		; vorherige Verarbeitungsebene.	!	INC	HL
	((RP))→IP	; IP zeigt ins Leere (Ende des Secondary) und wird mit	!	LD B	,(HL)
		; dem letzten geretteten Wert (nächste physische	!	INC	HL
		; Adresse der vorherigen Verarbeitungsebene) geladen.	!	LD (RPP)	,HL
	(RP) ;= RP−2	; Rückstellen von RP.	!	JP	NEXT
	JUMP NEXT	; NEXTes Wort in vorheriger Verarbeitungsebene.	!		
PFADUP:		; Maschinencode des Wortes DUP (DUPliziere TOS)	! DUP:		
		; befindet sich im Parameterfeld von DUP (2 Byte	!	POP	DE
		; unter der CFA beginnend).	!	PUSH	DE
	((DP)+2) ;= ((DP))	; Kopiere TOS in nächste RAM-Zelle.	!	PUSH	DE
	(DP) := (DP) +2	; Setze Zeiger auf TOS neu.	.	JP	NEXT
	JUMP NEXT	; Fertig zum NEXTen Wort.	!		

(x)	-bezeichnet Inhalt von x	!	(IP) = BC – Doppelregister
→	-symbolisiert eine Ladeoperation	!	DP = SP – CPU-Stackpointer
:=	-symbolisiert eine Neuzuweisung	!	(W) = HL – Doppelregister
		!	T = DE – Doppelregister

Anhand der Routinen NEXT, DOCOL und SEMIS läßt sich gut verfolgen, wie der FORTH-Prozessor durch die einzelnen Verarbeitungsebenen zu den Primitive-Worten gelangt, diese ausführt (DUP steht als ein Beispiel) und letztlich in die höchste rufende Ebene zurückkehrt. Dies ist i. allg. der äußere Interpreter, die Endlosschleife **QUIT**, dessen compiliertes SEMIS nie erreicht wird. Zur Verdeutlichung der ablaufenden Prozesse sind die entsprechenden Code-Teile eines U880-Assemblerlistings mit dargestellt. Der modulare Aufbau von FORTH ist auch hier im Systemkern gut zu erkennen. Es bereitet kaum Mühe, die Codierung zu verstehen. Damit ist eine wesentliche Voraussetzung für eigene Modifikationen, Erweiterungen oder Neuimplementierungen gegeben. Die Wechselbeziehungen der einzelnen Register sind in /1/ ausführlicher dargestellt.

4. Schlußbetrachtung

Es ist nicht möglich, innerhalb eines Artikels auf alle Aspekte dieses Softwarewerkzeuges einzugehen. So mußten Betrachtungen zu verschiedenen verfügbaren Hilfsmitteln wie Discompiler und Tracer, Meta- und Cross-Compiler (z. B. /17/) oder turn-key-Funktionen ausgespart werden. Vektorielle Execution und Vorwärtsreferenzen seien ebenfalls nur genannt.

FORTH ermöglicht den Zugriff auf ausnahmslos alle Systemressourcen bei gleichzeitiger Verfügbarkeit als Hochsprache. Daraus resultiert eine erhöhte Verantwortung, denn das System erkauft diesen Komfort mit einer recht dürftigen Fehlerbehandlung. So gibt es z. B. keine Datentypenüberwachung wie etwa in der Sprache C. Aber auch hier kann bei Bedarf erweitert werden.

FORTH ist nicht leicht zugänglich. Der Einarbeitungsaufwand ist mit Sicherheit höher als z. B. bei Pascal. Der Anwender wird dafür jedoch voll entschädigt, wenn er einmal die Handhabung dieses Werkzeuges beherrscht und die dahinter stehende Philosophie /18/ verstanden hat.

FORTH-Quelltext ist nicht unbedingt leicht zu lesen. Die Zeit für eine hinreichende Dokumentation über jeden geschaffenen Modul sollte man im eigenen Interesse investieren. Rekapitulation ist bekanntlich recht mühselig.

Anwendungen sind u. a. auf dem Gebiet der Robotertechnik /19/, /20/ bekannt geworden. Das modulare Konzept, die Transparenz und die permanente Regenerierungsfähigkeit von FORTH-Systemen macht FORTH auch für die Mitgestaltung großer und größter Softwarepakete prädestiniert.

In FORTH Programmieren impliziert das systematische Aufbereiten der Aufgabenstellung. Gleichzeitig bleiben dem Programmierer viele Freiräume bei der Realisierung der einzelnen Komponenten. FORTH zeichnet sich durch sehr kurze Wechsel zwischen Codierung und Test aus. Diese interaktive Programmiertechnik ermöglicht schnelle Fehlerbeseitigung und hohe Produktivität /22/.

FORTH ist eine junge, noch im Wachsen begriffene Programmiersprache. Obwohl von Natur aus einer Standardisierung entgegenstehend, sind bisherige Bemühungen der Forth Interest Group (figFORTH, FORTH83) recht erfolgreich. Anpaßprobleme von Programmen an unterschiedliche FORTH-Versionen gibt es praktisch nicht, solange die gängigen Systemschnittstellen eingehalten werden. Fehlende Worte können schnell hinzu definiert werden.

Die bisher relativ geringe Akzeptanz von FORTH wird nicht unbedingt auf den höheren Einarbeitungsaufwand gegenüber anderen Programmiersprachen zurückgeführt, sondern ist in unzureichender Information über diese Programmiertechnik begründet. Diesem Umstand etwas zu begegnen, war Anlaß, diesen Artikel zu schreiben.

Literatur

/1/ Zech, R.: **Die Programmiersprache FORTH.** Franzis Verlag München 1984

/2/ Glasmacher, P.: FORTH in Silizium. c't Magazin für Computertechnik Heft 4/1987, S. 36–39

/3/ Forth direkt ausgeführt. mc Mikrocomputerzeitschrift Heft 6/1985, S. 63

/4/ Vack, G. U.: Hardware-Realisierung von FORTH. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 169, 170

/5/ Odete, L.: Z8000 Forth. Dr. Dobb's Journal, Nr. 71, Sept. 1982, S. 48–63

/6/ Schiemann, B.: Ein fig-kompatibles FORTH für den U8000. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 165, 166

/7/ Woitzel, E.: comFORTH – Ein flexibles Programmierwerkzeug zur Prozeßautomatisierung. Industriebeleg WPU Rostock 1985

/8/ Zech, R.: FORTH 83. Franzis Verlag München 1987

/9/ Brodie, L.: Programmieren in FORTH. Carl Hanser Verlag München 1986

/10/ Monroe, A. J.: Forth Floating-Point Package. Dr. Dobb's Journal, Nr. 71, Sept. 1982, S. 16 bis 29

(Fortsetzung auf Seite 44)

Universelles 3D-Grafikprogramm in einer Anwendung zur 2dimensionalen Schnellen Fourier-Transformation

Bodo Bachmann
VEB Mikroelektronik „Karl Marx" Erfurt

In vielen Fällen ist es günstig, eine Vorstellung vom Verlauf mathematischer Funktionen bzw. experimentell ermittelter Meßwerte zu gewinnen. Im folgenden wird ein Programm vorgestellt, welches Funktionen der Form $z = f(x,y)$ räumlich, perspektivisch darstellt, wobei die grafischen Darstellungen mathematisch verdrehbar und somit aus beliebigen Richtungen betrachtbar sind. Daran anschließend wird ein Programm zur Schnellen Fourier-Transformation beschrieben, dessen Transformationspaare wirkungsvoll mit dem 3D-Grafikmodul dargestellt werden können.

3D-Darstellung

Um ein 3dimensionales Bild entsprechender Auflösung mit einem Mikrorechner auch in kurzer Zeit darzustellen, wurde die Programmiersprache FORTH verwendet. Dabei kamen die in /2/ und /3/ beschriebenen Erweiterungen des figFORTH-Standards zur Anwendung. Der Plotter robotron K 6418 diente als Zeichengerät. Es kann aber auch bei geringer Modifikation des Programmes ein Grafikdisplay angesteuert werden.
Das Programm setzt sich aus folgenden Hauptteilen zusammen:

1. Koordinatentransformation
2. Projektion (Parallelprojektion)
3. Zeichnen mit Weglassen der verdeckten Linien.

Die Theorie dazu ist ausführlich in /1/ beschrieben.
Vorausgesetzt wird eine Matrix von Funktionswerten des darzustellenden Objektes im x,y,z-Koordinatensystem. Durch die Koordinaten x und y wird eine Speicheradresse eindeutig charakterisiert. Auf dieser Adresse steht dann der Funktionswert z. Die Matrix wird also als Kette von Funktionswerten im Speicher abgelegt.
Mit den gewünschten Betrachtungswinkeln (Drehung und Kippung) transformieren sich die Koordinaten x,y,z zu den Koordinaten u,v,w.
Es werden gleich am Anfang die Schrittweiten in x- und y-Richtung transformiert, da dann in einem kartesischen Raster jeder Punkt durch deren Vektoraddition erreicht werden kann. Der Programmteil für die Transformation der y-Schrittweite ist in Bild 1 zu sehen. Es werden die transformierten Werte UY und VY, die die Schrittweite zum nächsten Punkt charakterisieren, auf dem Parameterstack (PS) bereitgestellt.
Das Programm entspricht den Formeln

$$UY = -\cos(\alpha)\,2000/(n-1)\,\mathrm{SGN}\,(\sin[\alpha]) \quad (1)$$

$$VY = \sin(\alpha)\sin(\beta)\,2000/(n-1)\,\mathrm{SGN}\,(\sin[\alpha]) \quad (2)$$

α bezeichnet den Drehwinkel, β den Kippwinkel, N die Dimension (max. 81), und 2000 ist dabei ein Normierungsfaktor. Schon hier ist der Vorteil eines extra Gleitkommastacks (GS) zu ersehen. So spart man sich einige SWAP-, ROT- oder OVER-Befehle.
Die x-Schrittweite wird analog transformiert. Nun kann die erste Matrixzeile in einen Zwischenspeicher geholt werden. Mittels des in Bild 2 gezeigten Wortes PROJ wird dann jeweils ein einzelner Punkt projiziert. Zu übergeben sind dabei auf dem Parameterstack UY, VY, LU, LV, I und auf dem Gleitkommastack $\cos(\beta)$.
LU und LV bezeichnen den aktuellen Stand des Plotters, I den Punktezähler.
Man erhält auf dem Parameterstack UY, VY (unverändert), LU', LV', U, V und auf dem Gleitkommastack wieder $\cos(\beta)$. U und V sind die vom Plotter anzufahrenden Koordinaten. Zu entscheiden wäre noch, ob die Linie auch gezeichnet wird. Das gleich an PROJ anzuschließende Wort LINE (Bild 3) überprüft den aktuellen Punkt mit den Koordinaten U und V, ob er evtl. durch andere Punkte verdeckt wird. Ist dies nicht der Fall, wird die Feder des Plotters gesenkt und die Linie bis zum Punkt gezogen, sonst bleibt die Feder angehoben.
Das Prinzip des Nichtzeichnens verdeckter Bildlinien beruht darauf, daß Bildlinien von Linien des Objektes, die dem Betrachter näher liegen, früher gezeichnet werden als die Bildlinien von Objektlinien, die weiter entfernt sind, und daß jede Bildlinie, die gezeichnet werden soll, mit einem unteren Grenzvektor verglichen wird. Sie wird nur dort gezeichnet, wo sie größer als der obere oder kleiner als der untere Grenzvektor ist. Beide Vektoren werden gleichzeitig auf den neuesten Stand gebracht, das heißt, die Bildlinie wird in die Grenzenbildung mit einbezogen.
Im Wort LINE werden die Adressen der Vergleichspunkte berechnet, und deren Werte werden mit V verglichen. Ist der entsprechende Vergleichspunkt größer bzw. kleiner, so wird die Variable T inkrementiert. Nur wenn T den Wert 2 erreicht und wenn zusätzlich T beim vorherigen Punkt auch 2 erreicht hat (wird in 2T gemerkt), wird die Feder gesenkt. Jetzt werden die Koordinaten U und V für den Plotter normiert, erhalten einen Offset und können an den Plotter übergeben werden.
Es fehlen nun noch einige Rahmenwörter, die spezielle Winkelbereiche testen und die Schleifen zum Punktedurchlauf definieren. Diese sind relativ einfach und werden deshalb nicht extra erläutert.
Die Bilder 4 bis 6 zeigen einige Beispiele für mit diesem Programm gezeichnete Funktionen, deren Auflösung 41 mal 41 Punkte beträgt. In den Bildern 4 und 5 wurde allerdings nur jede zweite Linie gezeichnet. (Die Grafik aus Bild 5 verwendeten wir wegen ihrer Attraktivität bereits zur Gestaltung des

```
: Y-TRAFO                (2 od. 1 -- UY VY)
-2000. N@ 1 - *          ( (N-1)·2,wenn nur jede 2. Linie
                          gezeichnet wird                )
S->G                     (Zahl vom PS zum GS)
G/ VSA G@                (VSA hält das Vorzeichen von sinα )
G* GDUP CA G@            (CA hält als Variable cosα )
G* W G@ G*               (W=Maßstabsfaktor )
G->S SA G@ SB G@         (Zahl vom GS zum PS sinα, sinβ )
G* G* GVZ                (GVZ = Vorzeichenwechsel )
W G@ G* G->S  ;
```

Bild 1 FORTH-Wort zur Schrittweitentransformation

Bild 2 Berechnung der Projektionskoordinaten eines Punktes

Bild 3 Zeichnen ohne verdeckte Linien

```
: PROJ                   PS (I UY VY LU LV -- UY VY LU' LV' U V)
                         GS (       CB     --      CB         )
41700 + @                   ( z vom Zwischenspeicher          )
GDUP S->G G*                (Multiplikation mit CB            )
SWAP DUP
5 PICK                      ( 5. Element auf oberste Stackpos.)
+ SWAP ROT DUP
5 PICK + ROT ROT
G->S +        ;
```

```
: LINE                     ( U V -- )
0 T ! 2DUP SWAP S->G
D G@                       (D=Gleitkommavariable für Differenz
                            der U-Koordinaten zweier
                            benachbarter Punkte              )
G/ 0.5 G+                  (Offset zum Vergleichsvektor      )
G->S                       (Integer für Adreßrechnung        )
2* DUP 41070               (Multiplikation mit 2, da 2-Byte-
                            Zahl                             )
+@ ROT <                   (1. Test auf Verdeckung           )
IF 2DUP 41070 + !
ELSE 1 T +!
ENDIF
2DUP 41490 + @ <           (2. Test auf Verdeckung           )
IF 2DUP 41490 + !
ELSE 1 T +!
ENDIF
DROP T @ 2 <
IF 2T @ 2 < IF PD;         (2T und T = 2, also Linie ziehen  )
            ELSE PU;
            ENDIF
ELSE PU;
ENDIF
SWAP 5 + 10 / 01 @ +       (Normieren und Plotter Offset     )
SWAP 5 + 10 / 02 @ +
PA T @ 2T !      ;         (T in 2T für nächsten Punkt merken )
```

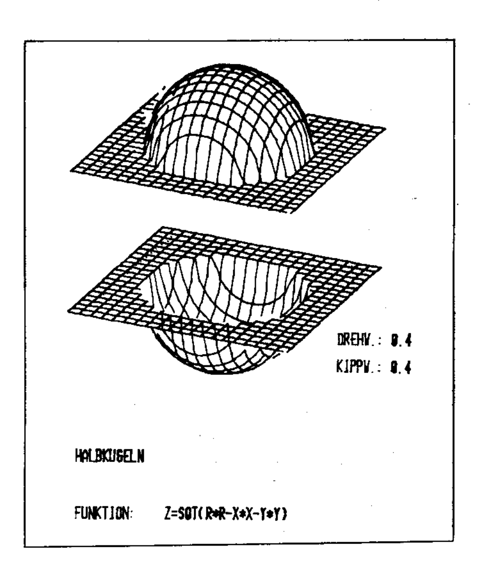


Bild 4 Halbkugelpaar

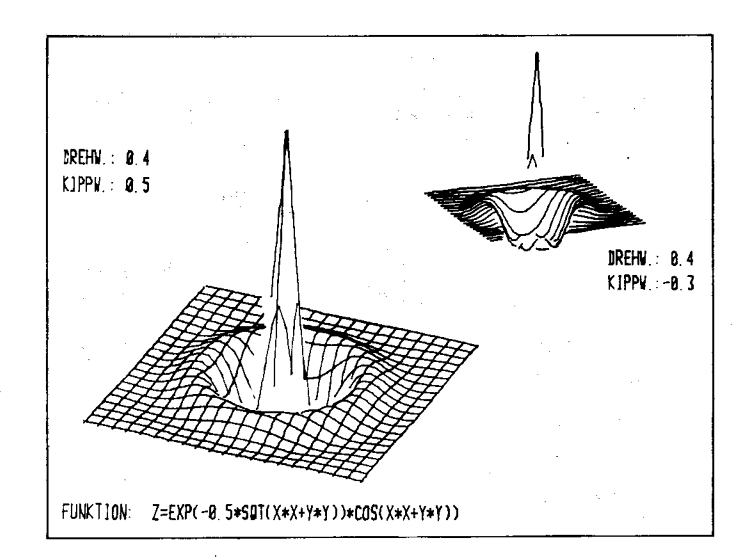
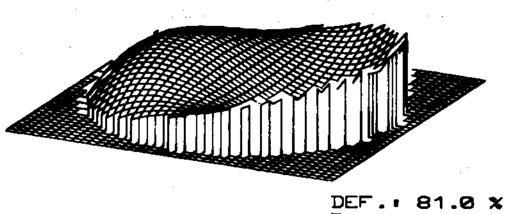


Bild 5 Beispiel aus zwei verschiedenen Betrachtungsrichtungen

Titelbildes von MP 11/1987, als es um das Thema Computergrafik ging.)
Im Programm können maximal 81 mal 81 Punkte dargestellt werden.
Interessant ist der Zeitvergleich zu einem entsprechenden BASIC-Programm, welches für die Dimension 41 ca. 20 min benötigt. Das FORTH-Programm hingegen braucht nur 2 min, wobei, wie auch in /3/ schon angedeutet, diese Zeit durch die Zeichengeschwindigkeit des Plotters begrenzt wird. Die reine Rechenzeit liegt noch darunter. (Die Zeiten beziehen sich auf den Bürocomputer A 5130 mit 2,5 MHz Taktfrequenz.)

Schnelle Fourier-Transformation

Die 2dimensionale Fourier-Transformation basiert hier im dargestellten Programm auf dem Schnellen Fourier-Transformationsalgorithmus nach Cooley und Tukey, speziell unter Verwendung der Basis 2 /4/. Repräsentiert wird dieser Programmteil durch das FORTH-Wort SFT (Bild 8). Ein analoges BASIC-Programm, dessen Funktionsweise sicher leichter nachzuvollziehen ist als das FORTH-Wort SFT, ist in /4/ zu finden. Jedoch weist das Wort SFT noch spezielle programmtechnische Besonderheiten auf. So werden beispielsweise nur 2-Byte-Integerzahlen verarbeitet. Die Gleitkommarechnung wird hier nur zur Berechnung der Winkelfunktionswerte verwendet. Durch die Integerrechnung sind hohe Rechengeschwindigkeiten zu erzielen. Zur weiteren Steigerung der Rechengeschwindigkeit werden die benötigten Winkelfunktionswerte durch das Wort COSI im Speicher bereitgestellt.
Die in SFT benötigte Bitspiegelung (INVERS) sowie eine besonders schnelle Multiplikationsroutine (MS∗) wurden als FORTH-Primitive definiert (Bild 7).
Im Wort SFT wird ein Vektor von N Elementen mit Real- und Imaginärteil transformiert. N ist dabei im Rahmen der 2er Potenzen bis 64 frei wählbar und zu Beginn der Rechnung in die Variable NF zu laden. Bei einer anderen Speicheraufteilung sind auch höhere Potenzen möglich.
Die Konstante ROFF enthält die Adresse des ersten Elementes des Realteil-Vektors. 128 Byte höher im Speicher wird durch die Konstante IOFF der Platz für den Imaginärteil des ersten Elementes markiert.
Zur Erläuterung werden in Bild 8 teilweise die Stack- bzw. Returnstack-Belegungen in Klammern dargestellt. Dabei bezeichnen speziell R1, R2, I1 und I2 die Adressen der gerade zu berechnenden Real- bzw. Imaginärteil-Elemente. Das Problem der Multiplikation mit den Winkelfunktionswerten wurde folgendermaßen gelöst: Die Winkelfunktionswerte, bereits bereitgestellt im Wort COSI, werden dort auf 32767 normiert. Zur Multiplikation wird dann die gemischt genaue Operation MS∗ verwendet, die aber nur die höherwertigen 2 Byte des doppelt genauen Ergebnisses auf dem Stack ablegt. Dies entspricht einer Division durch 65534, und durch die



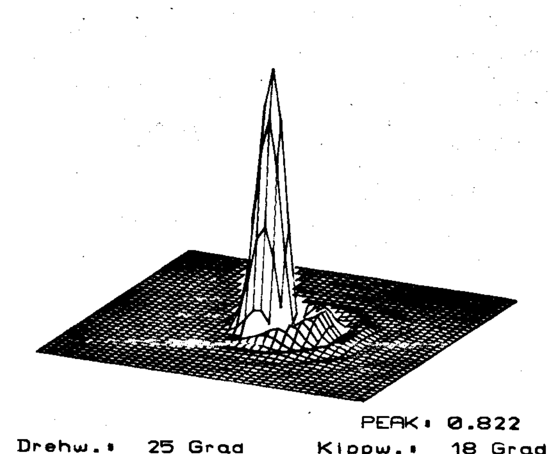


Bild 6 Anwendung in der Optik: Wellenfläche und Punktbild für Koma

Bild 7 Parameter und Maschinendefinitionen zur SFT

```
( ==== SFT/BBD ==== )
DECIMAL
   64 VARIABLE NF      0 VARIABLE CAF   62 ALLOT
    0 VARIABLE SAF     62 ALLOT
    0 VARIABLE IZ      0 VARIABLE T2     0 VARIABLE N2
    0 VARIABLE N5 48896 CONSTANT ROFF  49024 CONSTANT IOFF
HEX
CODE INVERS
   E1D1 , 7DC5 , 0806 , 27CB , 1DCB , FA10 , 3DCB , 23CB ,
   FA20 , E5C1 , 41C3 , 02 C,   END-CODE
CODE MS*
   E1D1 , 44C5 , 7A4D , E6A8 , F580 , 7ACB , 0628 , 21 , ED00 ,
   EB52 , 78CB , 828 , 21 , AF00 , 42ED , 4D44 , 21 , 3E00 ,
   CB11 , CB18 , 3D19 , 928 , 130 , CB19 , CB1C , 181D , F1F0 ,
   7FCB , F28 , 91AF , 3E4F , 9800 , 3E47 , 9D00 , 3E6F , 9C00 ,
   C167 , 40C3 , 02 C, END-CODE  DECIMAL                      -->
```

```
( ==== sft 2 ==== )

: SFT
  NF § N2 ! 0 N2 § 0 OVER 2* 2
  DO I IZ ! BEGIN SAF N5 ! DO
  N2 § +                                   ( KN2 )
  ROFF + DUP 128 + 2DUP                    ( R2I2R2I2 )
  I ROFF + DUP >R DUP 128 + DUP >R         ( R2I2R2I2R1I1  / R1I1)
  § ROT § DUP R> +! - T2 !                 ( R2I2R2R1  / R1 )
  § SWAP § DUP R> +! - >R                  ( R2I2  / T1 )
  N5 § DUP § DUP IF ROT X ! SWAP -72 + § ( R2 SIN COS  / T1 )
  OVER R MS* 2*   OVER T2 § MS* 2*   +  ( R2 SIN COS EI / T1)
  X § ! R> MS* 2*   SWAP T2 § MS* 2* - SWAP !
  ELSE 2DROP T2 § SWAP ! R> SWAP !
  ENDIF
                                                --->
```

```
( ==== sft 3 ==== )

  I2 § N5 +!

  I 2+ 2 +LOOP                             ( I )
  N2 § + DUP N2 § + OVER                   ( K K+N2 K )
  DUP NF § 2* = UNTIL 2DROP DROP
  0 N2 § 2 / DUP N2 ! 0                    ( 0 N2/2 0 )
  I +LOOP 2DROP DROP
  NF § 0 DO I NF § INVERS
  DUP I > IF 2* ROFF + DUP 128 +           ( R2I2 )
  I 2* ROFF + DUP 128 +                    ( R2I2R1I1 )
  ROT 2DUP § SWAP § ROT ! SWAP !
  2DUP § SWAP § ROT ! SWAP !
  ELSE DROP ENDIF LOOP ;
                                                --->
```

Bild 8 FORTH-Wort zur linearen Schnellen Fourier-Transformation

Bild 9 Weitere Hilfsfunktionen

```
( ==== sft 4 ==== )

: COSI
  3.1415926 NF § 2 / S->G G/
  32767 CAF ! 0 SAF !
  NF § 2 DO
  CAF I + 2 - § GDUP COS S->G G* G->S
  SAF I + 2 - § GDUP SIN GVZ S->G G* G->S -
  CAF I + !
  CAF I + 2 - § GDUP SIN GVZ S->G G* G->S
  SAF I + 2 - § GDUP COS S->G G* G->S +
  SAF I + !
  2 +LOOP GDROP ;

                                                --->
```

```
( ==== sft 5 ==== )

: NEGAT
  NF § 2* ROFF + ROFF 2+ DO I § MINUS I ! I 128 + DUP § MINUS
  SWAP ! 4 +LOOP ;
: TEIL
  NF § 2* ROFF + ROFF DO I § 5 + 10 / I ! I 128 + DUP § 5 +
  10 / SWAP ! 2 +LOOP ;
```

anschließende Multiplikation hat quasi eine Multiplikation mit dem echten Winkelfunktionswert (≦ 1) stattgefunden. Dabei wurde eine fast 4stellige Genauigkeit erzielt. Da alle anderen Operationen aus Additionen bestehen, trifft die Genauigkeit für das ganze Wort SFT zu. Am Ende des Wortes wird die notwendige Umordnung der Elemente mit Hilfe des Maschinenwortes INVERS (Bitspiegelung) vollzogen.
Um den Rahmen der 2-Byte-Integerzahlen nicht zu sprengen, muß darauf geachtet werden, daß die Summe der eindimensional zu transformierenden Elemente die Zahl 32767 nicht überschreitet. Die zu transformierenden Vektoren sind also entsprechend zu normieren.
Weiterhin ist zu beachten, daß die Definitionswerte nur positiv sein können und Funktionen mit negativen Definitionswerten entsprechend ihrer Periodizität verschoben werden müssen. Eine solche Verschiebung wird bei Funktionen mit einem Definitionsbereich von − x bis + x durch den Aufruf des Wortes NEGAT (Bild 9) bewirkt. In Bild 9 ist noch die Definition des Wortes TEIL gezeigt, welches nach der Fourier-Analyse jedes Element durch N teilt.
Für eine Rücktransformation (Fourier-Synthese) sind im Wort SFT in den beiden Zeilen, die auch das Wort MS* aufrufen, Addition und Subtraktion zu vertauschen. Das Wort TEIL wird dann nicht aufgerufen.
Die Erweiterung des Fourier-Transformationsprogramms zur Transformation quadratischer Matrizen, also Funktionen 2er Veränderlicher, erfolgt am effektivsten, indem zuerst alle Zeilen linear transformiert und danach die Spalten der so veränderten Matrix linear transformiert werden oder umgekehrt. Ein Beispiel dazu aus dem Bereich der Optik zeigt Bild 6. Dargestellt werden die durch ein angenommenes Objektiv, welches lediglich durch einen einzelnen Bildfehler (Koma) belastet ist, erzeugten Wellenaberrationen und die daraus durch eine Fourier-Transformation mit anschließender Intensitätsbildung (Quadratsumme von Real- und Imaginärteil) gewonnene Punktbildfunktion.
Bei jeder Fourier-Transformation ist besonders auf die Einhaltung des Abtasttheorems /4/ zu achten, dessen Berücksichtigung in diesem speziellen Fall ergab, daß die Stützpunktzahl der Wellenfläche maximal gleich der Hälfte der Dimension der Matrix sein darf. Die äußeren Zeilen und Spalten werden mit Nullen aufgefüllt. Auf diese Weise sind Bandüberlappungen ausgeschlossen, und auch die Forderung der Bandbegrenzung ist erfüllt.
Die Dimension der zu transformierenden Matrix des Beispiels aus Bild 6 betrug 64 mal 64 Punkte. Die zur Transformation notwendige Rechenzeit belief sich auf ca. 4 min. Mit dem 3D-Grafikmodul wurde jeweils nur ein Ausschnitt der Matrix von 41 mal 41 Punkten dargestellt.
Abschließend soll noch bemerkt werden, daß der Programmieraufwand bzw. die Entwicklungszeit für ein solches FORTH-Programm etwa 2- bis 3mal höher als für ein vergleichbares BASIC-Programm ist.

Literatur

/1/ Bechstein, J.: Perspektivische Darstellung von Funktionen zweier Veränderlicher mit Hilfe von Plotter und Display. Zentralinstitut für Kernforschung 1977
/2/ Bachmann, B.: Gleitkomma in FORTH. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 166
/3/ Bachmann, B.: Steuerung des Plotters K 6418 in FORTH. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 168
/4/ Brigham, E. O.: FFT Schnelle Fourier-Transformation. Einführung in die Nachrichtentechnik. R. Oldenbourg Verlag München, Wien 1985

☒ KONTAKT ☏

VEB Mikroelektronik „Karl Marx" Erfurt, Abt. CET5, Rudolfstr. 47, Erfurt, 5010; Tel. 583968

(Fortsetzung von Seite 41)

/11/ Bachmann, B.: Gleitkomma in FORTH. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 166–168
/12/ Woitzel, E.: comForth – Programmierwerkzeug FORTH unter SCP. edv-aspekte, Berlin 5 (1986) 4, S. 47–52
/13/ Solntseff, N.: Forth Programming Style. Dr. Dobb's Journal, Nr. 71, Sept. 1982, S. 30, 32
/14/ Zech, R.: Forth auf einem Einchip-Mikro. mc Mikrocomputer-Zeitschrift Heft 1/1985, S. 96 bis 100
/15/ Floegel, E.: FORTH Anwendungsbeispiele. Hofacker Verlag Holzkirchen 1984
/16/ Duncan, R.; Tracy, M.: The FVG Standard Floating Point Extension. Dr. Dobb's Journal, Nr. 71, Sept. 1984, S. 110–115
/17/ Buvel, R.: A Forth Native Code Cross Compiler for the MC68000. Dr. Dobb's Journal, Sept. 1984, S. 68–107
/18/ Brodie, L.: In FORTH denken. Carl Hanser Verlag München 1987
/19/ Bruijn, P. M.: Forth programming in control engineering. Journal A, Vol. 27, no 2, 1986, S. 91–96
/20/ Varga, G.; Krapp, M.: FORTH als Sprachkonzept für Robotersteuerungen. edv-aspekte, Berlin 4 (1985) 4, S. 11–15
/21/ Recherche Nr. 9397 vom 22. 7. 1986
• Thema: Programmiersprache FORTH
VEB Robotron-Elektronik Dresden, Leitstelle Inform./Dokumentation, Postfach 330, Dresden, 8012
/22/ Meyer, G.: SPS-Programmierung in FORTH. Markt und Technik, München, Heft 8/1984, S. 33, 34

Mikroprozessorsystem K 1810 WM86

Hardware · Software · Applikation (Teil 1)

Prof. Dr. Bernd-Georg Münzer (wissenschaftliche Leitung),
Dr. Günter Jorke, Eckhard Engemann, Wolfgang Kabatzke, Frank Kamrad, Helfried Schumacher, Tomasz Stachowiak
Wilhelm-Pieck-Universität Rostock, Sektion Technische Elektronik, Wissenschaftsbereich Mikrorechentechnik/Schaltungstechnik

Das System K 1810 WM86 wird in dieser Kurs-Reihe in folgenden Komplexen umfassend vorgestellt:

1. Systemarchitektur
2. System-Schaltkreise
3. Interface-Schaltkreise
4. Interruptsystem
5. CPU-Assemblerbefehle
6. Assemblerprogrammierung mit SCP 1700
7. Coprozessoren
8. Programmentwicklung in C
9. Multitask-Verarbeitung
10. Systemüberblick

1. Systemarchitektur

Das 16-Bit-Mikroprozessorsystem K 1810 WM86 (8086) wird durch folgende Leistungsmerkmale gekennzeichnet:

- Datenbreite 16 Bit
- Speicheradreßraum 1 MByte
- E/A-Adreßraum 64 KByte
- CPU-Registerbreite 16 Bit
  - 4 Hauptregister
  - 2 Pointerregister
  - 2 Indexregister
  - 1 Flagregister
  - 4 Segmentregister
  - 1 Programmzähler
- CPU-Takt 5 MHz
- Coprozessorfähigkeit.

Es entspricht somit der mittleren Klasse seiner 16-Bit-Generation.
Das System 8086 hat international eine außerordentlich breite Anwendung gefunden, besonders auch durch den Einsatz in Personalcomputern führender Hersteller.
In den einzelnen Kapiteln dieses Kurses werden die Architektur, System- und Interface-Schaltkreise im Rahmen von applikativ erprobten Anwenderlösungen vorgestellt. Des weiteren folgen Beschreibungen zum Befehlssatz und zu Adressierungsmodi mit Beispielen zur Assemblerprogrammierung. Ein weiterer Abschnitt beschäftigt sich mit dem Multi- und Coprozessorverhalten. Danach werden die Möglichkeiten zur Programmentwicklung mit Hochsprachen einschließlich der Echtzeitprogrammentwicklung erläutert. Das abschließende Kapitel gibt einen Ausblick zu weiterentwickelten Systemen der oberen 16-Bit-Klasse.
Der Kurs hat sich zum Ziel gesetzt, in der gesamten Breite von Hardware und Software

Der Kurs hat sich zum Ziel gesetzt, in der gesamten Breite von Hardware und Software die Erfahrungen zum System 8086 zu vermitteln. Bezüglich von Detailbeschreibungen zu Parametern der Systemkomponenten, Vollständigkeit des Befehlssatzes und von Dienstprogrammen der Betriebssysteme wird auf die entsprechende Literatur der Entwickler hingewiesen.

1.1. Systemkomponenten

Eine 8086-Mikrorechner-Grundkonfiguration in der praxisrelevanten CPU-Betriebsart Maximum-Mode besteht aus folgenden Komponenten (Bild 1.1):

Systemschaltkreise

– 8284A Clock-Generator: Generierung des Systemtaktes CLK, READY- und RESET-Steuerung.
– 8086 CPU: Generierung des multiplexen Adreß-/Datenbus AD0 ... AD15, der höherwertigen Adressen A16 ... A19, der Statussignale $\overline{S0}$ ... $\overline{S2}$ und der Speicher-Bank-Steuerung Bus-High-Enable $\overline{BHE}$.
– 8288 Buscontroller: Generierung der Steuersignale für Speicher-($\overline{MRDC}$, $\overline{MWTC}$) und ($\overline{IORC}$, $\overline{IOWC}$), des Strobe-Signals ALE zur Adreßübernahme, der Freigabe- und Richtungssteuerung für die Datenbustreiber (DEN, DT/$\overline{R}$) und des Signales $\overline{INTA}$ zur Interruptbestätigung.

1.2. Busstruktur

Die 8086-CPU besitzt einen multiplexen Adreß-/Datenbus, der über Octal-Latches 8282 zum System-Adreßbus AB0 ... AB19 und über Datenbustreiber 8286 zum System-Datenbus DB ... DB15 geführt wird. Die Selektion von Daten und Adressen erfolgt mit Hilfe der System-Steuersignale

ALE address latch enable
DEN data enable
DT/R data transmit/receive.

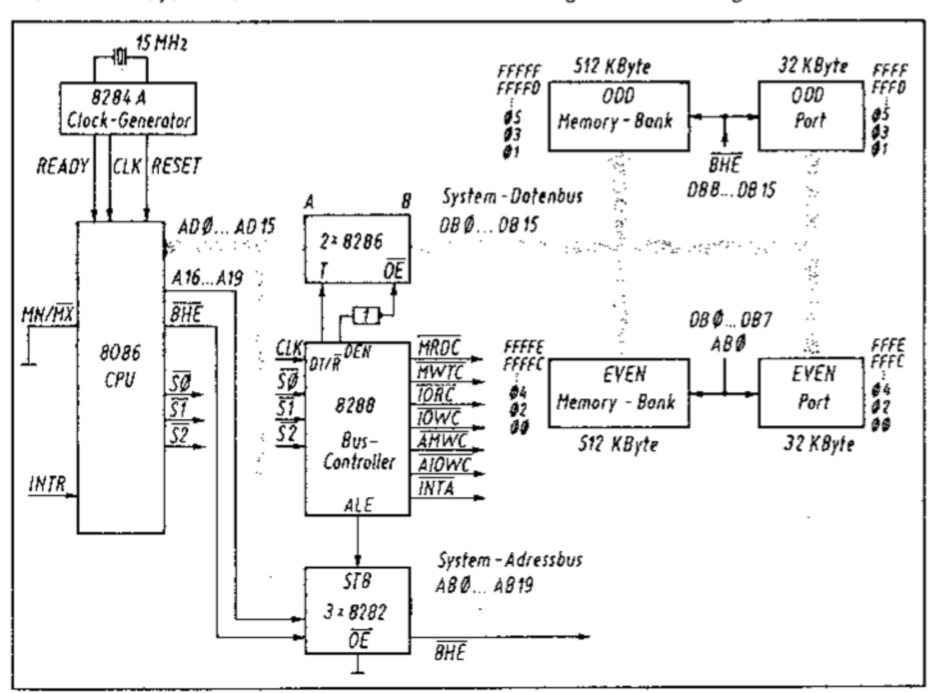
1.3. Speicher und E/A-Ports

Der Arbeitsspeicher ist byteorganisiert aufgebaut, das heißt, der gesamte Adreßraum von max. 1 MByte ist in 2 physisch getrennte 512-KByte-Banks (EVEN-, ODD-Bank) aufgeteilt. Die EVEN-Bank repräsentiert alle geraden Adressen (Chip-Select mit AB0 = 0), und die 8-Bit-Datenleitungen der Speicherbank sind zum niederwertigen Teil DB0 = DB7 des Systembusses geführt. Dementsprechend vereinigt die ODD-Bank alle ungeraden Adressen (Chip-Select mit $\overline{BHE}$ = 0), deren Datenleitungen mit DB8 ... DB15 verbunden sind. Die 8086-Ports mit dem maximalen Adreßraum von 64 KByte sind ebenfalls byteorganisiert und werden durch gerade/ungerade Adressen mittels AB0 und $\overline{BHE}$ selektiert (Bild 1.1). Die Ein-/Ausgabebefehle erlauben somit neben der Byte-Ein-/Ausgabe auch den Worttransfer über zwei 16-Bit-Ports benachbarter Adressen.

1.4. CPU Basis-Timing

Der 5-MHz-Systemtakt CLK, vom 8284A bereitgestellt, ist unsymmetrisch mit 1/3 High-Pegel und 2/3 Low-Pegel (Bild 1.2). Jeder CPU-Buszyklus besteht aus mindestens 4 Takten.
Vor dem Takt T1 jedes Buszyklus erfolgt die Bereitstellung der Statussignale S0, S1, S2 nach folgender Kodierung:

Bild 1.1. 8086-Systemarchitektur



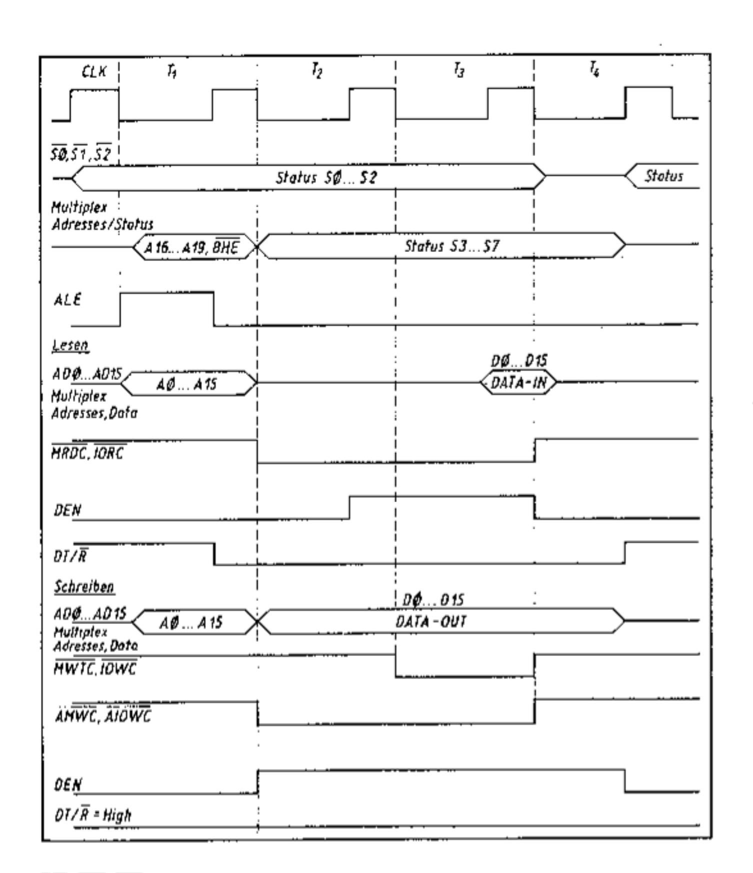


Bild 1.2 8086-Buszyklus

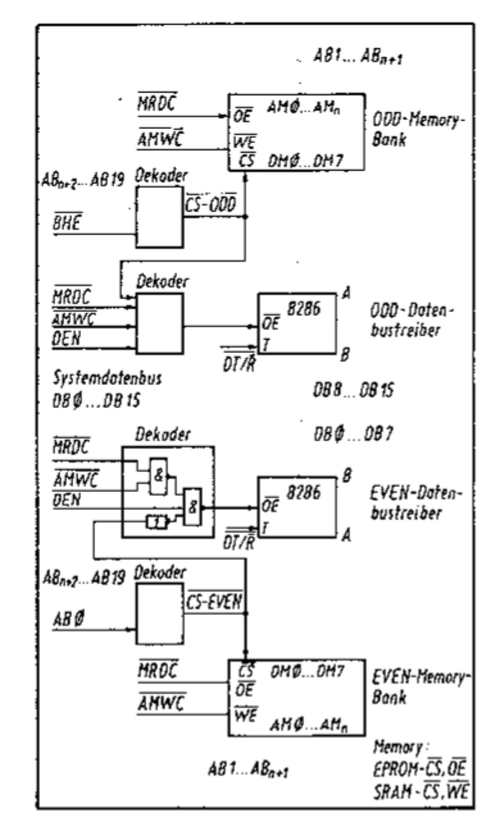


Bild 1.3 8086-Speicherarchitektur

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	
0	0	0	Interrupt-Bestätigung
0	0	1	Porteingabe
0	1	0	Portausgabe
0	1	1	Halt
1	0	0	Befehlholen
1	0	1	Speicherlesen
1	1	0	Speicherschreiben
1	1	1	passiv, kein Buszyklus.

Aus diesen Statussignalen erzeugt der Buscontroller 8288 die Steuersignale

$\overline{MRDC}$ memory read control
$\overline{IORC}$ I/O read control
$\overline{MWTC}$ memory write control
$\overline{IOWC}$ I/O write control

und die zeitlich vorgezogenen Schreibsignale

$\overline{AMWC}$ advanced memory write control
$\overline{AIOWC}$ advanced I/O write control.

Zur Adreßübernahme in die Octal-Latches dient das Signal ALE zur Richtungssteuerung der Datenbustreiber DT/$\overline{R}$ und zur Freigabesteuerung der Datenbustreiber DEN.

Die Statussignale S3 ... S7 können durch eine extreme Logik gespeichert werden und enthalten Informationen über aktuelle Zugriffe zu Segmentregistern und zu den Zustandsbits.

Im Takt T1 werden von der CPU die Adressen A0 ... A15 (multiplex mit Daten), A16 ... A19, $\overline{BHE}$ (multiplex mit Status S3 ... S7) generiert und von den Octal-Latches mit der fallenden Flanke von ALE übernommen. Zu Beginn von T2 stehen damit gültige Adressen und $\overline{BHE}$ bereit. Der Datentransfer wird durch Generierung der Steuersignale für die Speicher, für die E/A-Geräte und für die Datenbustreiber eingeleitet und mit den Takten T3/T4 abgeschlossen.

Speicherlese-/Befehlshole- und Portlesezyklen werden mit der Datenübernahme am Ende von T3 abgeschlossen. Bei Speicherschreib- und Portschreibzyklen werden unmittelbar nach T1 die Daten von der CPU bereitgestellt, um eine sichere Datenübernahme zu gewährleisten.

1.5. Speichertransfer

Der Datenaustausch zwischen CPU und Arbeitsspeicher erfolgt durch Aktivierung der Signale AB0, $\overline{BHE}$ byte- oder wortweise nach folgendem Modus:

$\overline{BHE}$	AB0	Transfer
1	0	Byte von gerader Adresse (EVEN-Bank) über DB0 ... DB7
0	1	Byte von ungerader Adresse (ODD-Bank) über DB ... DB15
0	0	Worttransfer gleichzeitig von EVEN- und ODD-Bank.

Befehlsholezyklen erfolgen grundsätzlich wortweise.

Die Adresse AB0 entspricht somit einem Freigabesignal für die EVEN-Bank (Bus-Low-Enable). Der 16-Bit-Transfer erfolgt mit AB0 = $\overline{BHE}$ = 0 parallel in einem Buszyklus, falls die niederwertige Adresse des Bytepaares geradzahlig ist. Wenn diese ungeradzahlig ist, wird der Worttransfer automatisch von der CPU in zwei Buszyklen zu je einem Byte zerlegt, zuerst mit AB0 = 1, $\overline{BHE}$ = 0 der ODD-Transfer und anschließend mit AB0 = 0, $\overline{BHE}$ = 1 der EVEN-Transfer.

Bild 1.3 zeigt einen Speichermodul mit EPROM und SRAM. Die Speicheradressen AM0 ... AMn werden mit AB1 ... ABn+1 des Systemadreßbus und die Datenleitungen DM0 ... DM7 jeweils mit dem niederwertigen Teil DB0 ... DB7 bzw. dem höherwertigen Teil DB8 ... DB15 des Systemdatenbus verbunden.

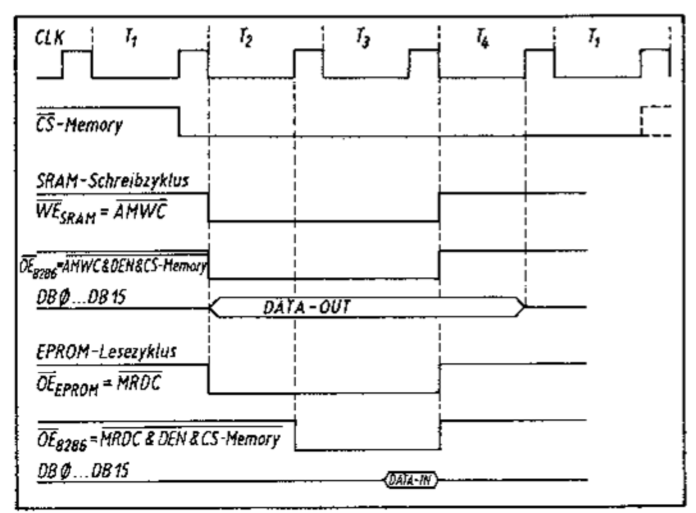


Bild 1.4 Speicherschreib-/-lesezyklus

Zur Sicherung der Datenzugriffszeiten wird die Speicher-CS-Steuerung nur aus den Adressen ABn + 2 . . . AB19 gebildet und für die EVEN/ODD-Bank mit AB0 bzw. $\overline{BHE}$ getort. Damit ist CS-Memory am Ende des Taktes T1 gültig (Bild 1.2). Im System 8086 dürfen nur Speicherschaltkreise mit pegelgesteuertem CS eingesetzt werden. Bei EPROMs wird der $\overline{OE}$-Eingang mit $\overline{MRDC}$ und bei SRAMs der $\overline{WE}$-Eingang mit $\overline{AMWC}$ verbunden.
Die Dekodierung für die OE-Freigabe-Steuerung der Datenbustreiber ist in Bild 1.3 im EVEN-Dekoder ausgeführt.
Die Richtungssteuerung T der Datenbustreiber erfolgt mit $DT/\overline{R}$.
Die Zeitbeziehungen beim Datentransfer am Beispiel eines SRAM-Schreibzyklus und eines EPROM-Lesezyklus sind in Bild 1.4 dargestellt.
Folgende Speicherschaltkreise können ohne WAIT-Zyklen bei einem Systemtakt von 5 MHz verwendet werden:

EPROM U2716 C 35
SRAM U2114 D 20.

Über die Anwendung dynamischer Speicher wird in einem späteren Abschnitt berichtet.
Eine praktikable Aufteilung des physischen 1-MByte-Speicherraumes in einem 8086-Rechner erfolgt nach folgendem Modus:

ROM

16 KByte	FC000H . . . FFFFFH	Monitor
	FFFF0H	Einsprung nach RESET

RAM

1/2 MByte	0400H . . . 7FFFFH	Daten, Stack und Anwenderbereich
1 KByte	0000H . . . 03FFH	Interrupt-Pointer-Tabelle

1.6. Ein-/Ausgabetransfer

Der Ein-/Ausgabetransfer über 8-Bit-/16-Bit-Ports wird äquivalent zum Speichertransfer nach Bild 1.1 und 1.3 mit folgenden systemeigenen Interface-Schaltkreisen realisiert:

8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART)
8255A Programmable Peripheral Interface (PPI)
8253 Programmable Interval Timer (PIT)
8259A Programmable Interrupt-Controller (PIC).

Die drei zuerst genannten Schaltkreise wurden aus dem 8-Bit-System 8080 übernommen. Diese Interface-Ports können keinen vektorisierten Interrupt auslösen, so daß ein im Interruptverhalten an die 8086-CPU angepaßter 8-Ebenen-Interrupt-Controller 8259A entwickelt wurde. Der PIC darf nur als EVEN-Port betrieben werden, da der Interruptvektor über den niederwertigen Systemdatenbus DB0 . . . DB7 von der CPU eingelesen wird. Für diese Schaltkreise ist bei einem E/A-Zugriff jeweils ein Wait-Takt einzufügen.
Bei der Kopplung von U880-Interface-Schaltkreisen an das System 8086 ist eine Reihe von schaltungstechnischen Maßnahmen zur Kompatibilität des Steuerbus und des Interruptverhaltens erforderlich.

2. System-Schaltkreise

Die 8086-Systemschaltkreise sind:
–8086 – CPU
–8284A – Clockgenerator
–8288 – Buscontroller.

2.1. CPU

Die Standard-8086-CPU, mit 5 MHz Systemtakt und der Betriebsspannung + 5 V, wird in HMOS-Silicon-Gate-Technologie mit 29000 integrierten Transistoren in einem 40poligen DIL-Gehäuse hergestellt. (Bild 2.1).

2.1.1. CPU-Architektur

2.1.1.1. Funktionseinheiten
Die internen Funktionen des 8086-Mikroprozessors sind in zwei Einheiten aufgeteilt (Bild 2.2):
– Execution-Unit (EU)
– Bus-Interface-Unit (BIU).

Die EU enthält die grundsätzlichen Elemente einer CPU, wie Hauptregister, Arithmetic & Logic-Unit, Adreß- und Flagregister. Die EU führt die Dekodierung der von der BIU zwischengespeicherten Befehle durch. Nur die BIU besitzt Zugang zum externen 8086-Bus. Zur Erhöhung der Busbandbreite führt die BIU unabhängig von der EU ein vorausschauendes Holen von Befehlscodes und deren Zwischenspeichern in einem 6-Byte-FIFO-Instruction-Queue durch. Die Übergabe der Befehlscodes aus dem Instruction-Queue an die EU hängt von der Dekodierung und Ausführung des aktuellen Befehls ab, wobei die BIU dabei bemüht ist, den Instruction-Queue ständig gefüllt zu halten. Diese parallele und asynchrone Arbeit zwischen EU und BIU führt zu einer optimalen Busauslastung. Die bisher vom Anwender gewohnte sequentielle Folge von Maschinenzyklen auf einem Mikrorechnerbus in Korrespondenz mit den Maschinenzyklen der Befehle wird beim 8086-System verlassen (pipelining). Der aktuelle Transfer auf dem Systembus wird vom jeweiligen Zustand der EU (Befehlsdekodierung, -ausführung) und vom Instruction-Queue in der BIU (vorausschauendes Befehlsholen) bestimmt. Auf diese Tatsache muß der Anwender sich bei der Logikanalyse bzw. Busanzeige im READY/WAIT-Einzelschrittbetrieb einstellen.
Weiterhin erfolgt in der BIU die Generierung des 20-Bit-Adreßbus mit Hilfe der Segmentregister (vgl. Abschnitt 2.1.1.5.).

2.1.1.2. Hauptregister

Die CPU besitzt vier 16-Bit-Hauptregister, die vor allem bei arithmetisch/logischen und E/A-Operationen verwendet werden (Bild 2.3):
AX Accumulator
BX Base
CX Count
DX Data.
Diese Register sind auch jeweils als zwei 8-Bit-Register verwendbar und mit dem Index H bzw. L bezeichnet. Durch diese Aufteilung lassen sich auch Operationen mit 8-Bit-Operanden durchführen.

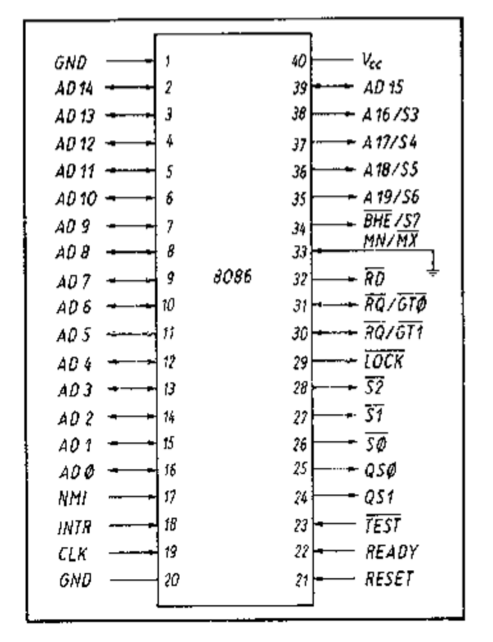


Bild 2.1 8086-CPU

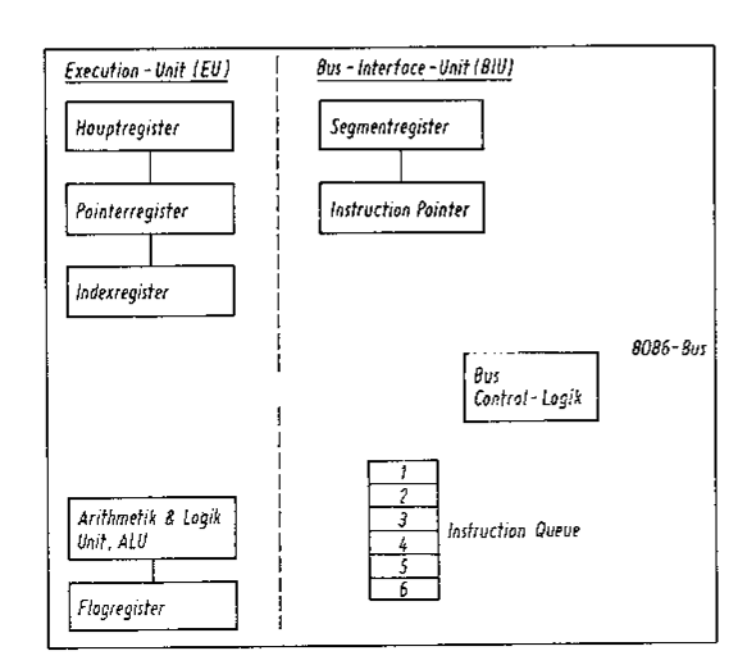


Bild 2.2 8086-CPU-Architektur

15			Ø ein 16 Bit-Register
	7 High Ø	7 Low Ø	zwei 8 Bit-Register
AX	AH	AL	Accumulator
BX	BH	BL	Base
CX	CH	CL	Count
DX	DH	DL	Data

Bild 2.3 Hauptregister

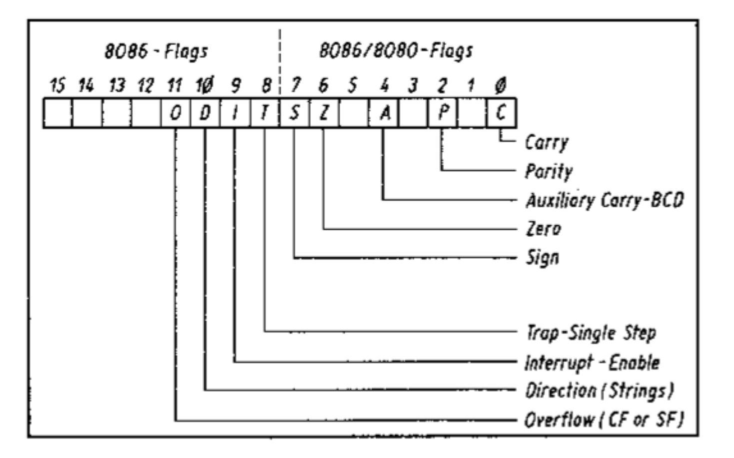


Bild 2.6 Flagregister

15 Ø	
SP	Stack Pointer
BP	Base Pointer
SI	Source Index
DI	Destination Index

Bild 2.4 Pointer- und Indexregister

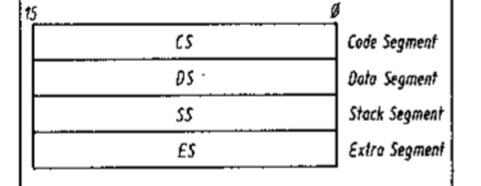


Bild 2.5 Segmentregister

Der Akkumulator AX wird vorzugsweise bei arithmetisch-logischen Operationen verwendet, und E/A-Befehle können nur über den Akkumulator ausgeführt werden. Das Base-Register BX dient zur Adressierung der Daten bei Speichertransfer-Operationen.
Das Count-Register CX wird bei LOOP- und Stringoperationen als Zählregister verwendet, und das Data-Register DX enthält bei einigen E/A-Operationen die 16-Bit-E/A-Adresse bzw. bei 16-Bit-Multiplikation/Division einen Teil des Ergebnisses.

2.1.1.3. Pointerregister

Über die 16-Bit-Pointerregister (Bild 2.4)

SP Stackpointer
BP Basepointer

werden Speicherplätze im Stacksegment adressiert. Der Stackpointer SP ermöglicht den Aufbau eines Stapels im Stacksegment. Mit dem Basepointer können zusätzliche Datentabellen im Stacksegment verwaltet werden.

2.1.1.4. Indexregister

Die 16-Bit-Indexregister (Bild 2.4)

SI Source-Index
SI Destination-Index

dienen zur Speicheradressierung bei Stringoperationen. Das SI-Register adressiert dabei den Quellbereich im Datensegment und das DI-Register den Zielbereich im Extrasegment.

Instruction-Pointer

Mit dem 16-Bit-Instruction-Pointer in der Bus-Interface-Unit erfolgt in Verbindung mit dem Code-Segment-Register die Adressierung von physischen Speicherplätzen bei Befehlsholezyklen.

2.1.1.5. Segment-Register

Die absolute Adressierung im 1-MByte-Speicherraum erfolgt mit Hilfe von 16-Bit-Segmentregistern (Bild 2.5):

CS Code-Segment
DS Data-Segment
SS Stack-Segment
ES Extra-Segment.

Jedes Segmentregister definiert einen 64-KByte-Bereich im Gesamtspeicherraum. Die Segmente müssen zu Beginn eines Programms geladen werden, und die Adreßbereiche können dabei auch überlappend festgelegt werden.
Das Code-Segment CS realisiert zusammen mit dem Instruction-Pointer die aktuelle Adresse des nächsten Befehls.
Das Stack-Segment SS realisiert z. B mit dem Stackpointer SP die aktuelle Adresse des Stapels bei PUSH, POP, CALL und RET-Operationen.
Mit dem Daten-Segment DS und z. B. dem BX-Register werden aktuelle Daten verwaltet.
Das Extra-Segment ES, als zusätzliches Datensegment, wird z. B. bei Stringoperationen eingesetzt.
Die Verwendung von Segmentregistern und Adreßregistern (IP, SP, BP, SI, DI, BX) erfolgt in definierten Zuordnungen, die im Abschnitt 2.1.3. erläutert werden.

2.1.1.6. Flagregister

Das 16-Bit-Flagregister (Bild 2.6) enthält im niederwertigen Teil 0...7 die aus dem System 8080 übernommenen Zustandsbits, deren Bedeutung denen im U880-System entsprechen (außer P-Flag). Im höherwertigen Teil 8...15 des Flagregisters sind folgende neue Zustandsbits aufgenommen.

T-Flag: Trap-Single Step

Dieses Zustandsbit wird beim Testen von Programmen eingesetzt und ermöglicht den softwaregesteuerten Einzelschrittbetrieb auch in ROM-Speicherbereichen. Wenn das T-Statusbit per Befehl auf 1 gesetzt wird, erfolgt am Ende des nächsten Befehls die Auslösung eines Software-Interrupts (vgl. Abschnitt Interrupt-Struktur).

I-Flag: Interrupt-Enable

Mit dem CPU-Befehl Interrupt-Enable wird das I-Flag = 1 gesetzt und damit der maskierbare Interrupt zugelassen. Nach einer Interruptannahme oder nach Ausführung des CPU-Befehls Interrupt-Disable erfolgt ein Rücksetzen des I-Flag und damit Interruptsperre.

D-Flag: Direction

Das D-Flag beeinflußt die Richtung des Datentransfers bei Stringoperationen. Bei Setzen des D-Flag = 1 mit dem CPU-Befehl Set-Direction erfolgt bei Stringoperationen automatisch ein Adressendekrement der Indexregister SI, DI. Mit dem CPU-Befehl Clear-Direction (D = 0) wird ein Auto-Inkrement von SI und DI realisiert.

O-Flag: Overflow

Bei arithmetisch-logischen Operationen wird das Carry-Bit in einer Exclusive-Oder-Verknüpfung mit dem Übertrag von Bit 6 nach Bit 7 zum Overflow-Flag verknüpft. Wenn das Overflow-Flag im Ergebnis einer Operation gesetzt und anschließend der Software-Interrupt-Befehl INTO ausgeführt wurde, erfolgt die Durchführung einer Interrupt-Service-Routine (vgl. Abschnitt Interrupt-Struktur).

2.1.2. Elektrische Signale und Anschlüsse

Die Anschlüsse der CPU (Bild 2.1) sollen nur in der Betriebsart Maximum-Mode (Anschluß MN/$\overline{MX}$ = 0) erläutert werden; die Betriebsart Minimum-Mode hat sich in der applikativen Praxis nicht durchgesetzt.
Die CPU-Signale können in die Kategorien
– Adreßsignale
– Datensignale
– Statussignale
eingeteilt werden. Eine Reihe der Anschlüsse ist zeitmultiplex ausgelegt.

Adreß- und Datensignale/Statussignale AD0...AD15 (input, output, tristate)

Im Takt T1 (vgl. Bild 1.2) werden bei Speicher- und E/A-Operationen die Adressen A0...A15 ausgegeben, in den nachfolgenden Takten T2 und T3 erfolgt der Datentransfer D0...D15.

A16/S3; A17/S4; A18/S5; A19/S6 (output, tristate)

An diesen 4 Anschlüssen werden bei Speicheroperationen im Takt T1 die höherwertigen Adressen A16 ... A19 gültig, bei E/A-Operationen sind diese Ausgänge low. Bei Speicher und E/A-Operationen sind während der Takte T2, T3, T4 die Statusinformationen S3, S4, S5 und S6 aktiv.
In S3 und S4 ist der Zugriff auf die Segmentregister im aktuellen Buszyklus nach folgender Vorschrift kodiert:

S4	S3	Bedeutung
0	0	Extra-Segment
0	1	Stack-Segment
1	0	Code-Segment oder kein Segment
1	1	Datensegment.

Die Statussignale enthalten folgende Angaben:
S5 Wert des Interrupt-Enable-Flag
S6 = 0; CPU ist aktueller Bus-Master
= tristate; CPU hat Buskontrolle abgegeben

$\overline{BHE}$/S7 (output, tristate)

Im Takt T1 erfolgt die Ausgabe des Speicherbank-Signales $\overline{BHE}$, der Status S7 in den Takten T2 ... T4 ist nicht näher definiert. Die Statussignale S3 ... S4 werden von den systemeigenen Schaltkreisen nicht ausgewertet.

$\overline{S0}$, $\overline{S1}$, $\overline{S2}$ (output, tristate)

Die Statussignale $\overline{S0}$, $\overline{S1}$, $\overline{S2}$ geben Informationen für den Buscontroller entsprechend der Kodierung von Abschnitt 1.

QS0, QS1 (output)

Die Statusbits QS0, QS1 werden von den Coprozessoren ausgewertet und geben Auskunft über den aktuellen Zustand im Zwischenspeicher des CPU-Instruction-Queue.

QS1	QS0	
0	0	keine Operation
0	1	erstes Byte eines Befehls wurde dem Instruction-Queue entnommen
1	0	Instruction-Queue ist leer
1	1	ein nachfolgendes Byte wurde dem Instruction-Queue entnommen

Diese Statusbits gelten während des Taktes nach einer Operation im Instruction-Queue.

Steuersignale
RESET (input, output, high-aktiv)

Das RESET-Signal, vom Clockgenerator 8284A einsynchronisiert, führt ein Rücksetzen der CPU durch. Danach stellt sich in der CPU folgender Zustand ein:
– Inhalt des Code-Segments = FFFFH
– Inhalt des Instruction-Pointers = 0000
damit lautet die physische Adresse des ersten Befehls FFFF0H
– Inhalt von Daten-, Stack- und Extra-Segment = 0000
– Rücksetzen aller Bits des Statusregisters (Interruptsperre, kein single-step).

INTR (input, high-aktiv)

Die pegelaktive Interruptanforderung wird im letzten Takt jedes Befehls abgefragt. Falls Interruptfreigabe gesetzt wurde und an INTR High-Pegel anliegt, erfolgt ein Interrupt-Annahmezyklus mit Generierung des Interrupt-Bestätigungssignals INTA durch den Buscontroller 8288 (vgl. Abschnitt Interrupt-Struktur).

NMI (input, high-aktiv)

Der flankengetriggerte NMI für den nichtmaskierbaren Interrupt wird im letzten Takt eines jeden Befehls abgefragt. Die Startadresse für die Interrupt-Service-Routine liest die CPU aus den Speicherplätzen 0008H ... 000BH.

READY (input, high-aktiv)

In Verbindung mit dem Clockgenerator 8284A erfolgt die WAIT-Steuerung der CPU (vgl. Abschnitt 2.2).

$\overline{TEST}$ (input, low-aktiv)

Nach Dekodierung eines WAIT-Befehls fragt die CPU den $\overline{TEST}$-Eingang ab. Es werden so lange keine weiteren Befehle eingelesen, bis $\overline{TEST}$ = low wird (vgl. Abschnitt Coprozessoren).

$\overline{RQ}/\overline{GT0}$, $\overline{RQ}/\overline{GT1}$ (input, output)

Die $\overline{RQ}/\overline{GT}$-Anschlüsse (0 = höhere, 1 = niedere Priorität) werden bei Anwesenheit weiterer Master im Local-Bus benutzt (DMA-Betrieb, Coprozessoren).
Die Anschlüsse sind bidirektional ausgeführt, und durch eine zeitlich genau definierte Folge von 3 Impulsen wird der Bus-Annahme- und Bus-Rückgabezyklus durchgeführt (vgl. Abschnitt Coprozessoren).

$\overline{LOCK}$ (output, tristate)

Ein Präfix LOCK vor einem Befehl bewirkt, daß während des nächsten Befehls der $\overline{LOCK}$-Ausgang aktiv = low wird. Damit wird anderen Coprozessoren mitgeteilt, daß während dieses geschützten Befehls keine Busübergabe stattfinden darf (vgl. Abschnitt Coprozessoren).

CLK (input)

Der CLK-Eingang der CPU ist mit dem entsprechenden Ausgang des Clockgenerators 8284A zu verbinden und realisiert den Systemtakt.

$\overline{RD}$ (output, tristate)

Ein Low-Pegel an diesem Ausgang signalisiert einen Speicherlese- oder Eingabezyklus. In Maximum-Mode wird dieses Signal kaum verwendet.

2.1.3. Speicheradressierung

Die Generierung der 20-Bit-physischen Speicheradresse im System 8086 erfolgt durch Addition von 2 Bestandteilen:
– 16-Bit-Basisadresse (Inhalt eines der Segmentregister CS, DS, SS, ES)
plus
– 16-Bit-Effektive-Adresse (Offset, Inhalt eines der Adreßregister IP, SP, BP, SI, DI, BX).

Die Bus-Interface-Unit BIU führt automatisch die Addition von Basis- und Offsetadresse nach folgendem Modus durch (Bild 2.7).

– Verschiebung des Inhaltes des Segmentregisters um 4 Bit-Positionen nach links, Auffüllung der Tetrade mit 0000
– Addition mit der Effektiven Adresse.

Ein Beispiel zur Realisierung einer konkreten physischen Adresse ist in Bild 2.8 dargestellt. Durch diese Aufteilung von Basis- und Offsetadresse wird der 1-MByte-Speicherraum in vier frei wählbare Segmente zu je 64 KByte

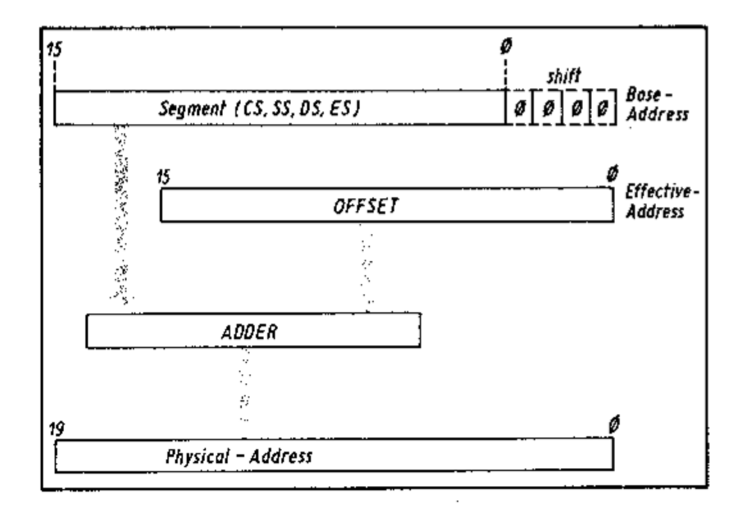


Bild 2.7 Adreßbildung mit Segment-Registern

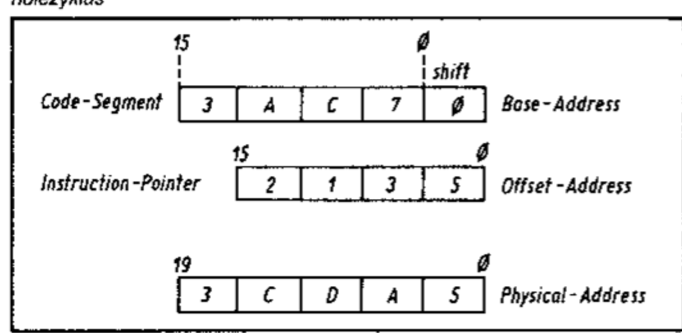


Bild 2.8 Beispiel zur Bildung der physikalischen Adresse bei einem Befehlsholezyklus

aufgeteilt. Die Segmentregister können eine beliebige Zahl enthalten und zeigen auf den Nullpunkt eines 64-KByte-Segment-Speicherraumes. Eine absolute Adresse ist nun innerhalb dieses Segment-Speicherraumes festgelegt. Der Abstand der Segment-Nullpunkte zueinander ist n × 16 Byte (n = 0,1 ... 64K). Durch Einführung der Segmentierung wird auch eine klare logisch-physische Trennung von Programmen (CS im ROM), Daten (DS und ES im RAM) und Stack (SS im RAM) erreicht. Im 8086-Befehlssatz mit seinen Adressierungsmodi ist eine Standard-Zuordnung der Register für die Bildung der Effektiven Adresse (Operanden-Register) mit den Segmentregistern festgelegt (Bild 2.9). Eine unveränderliche Zuordnung haben folgende Operandenregister:

- Instruction-Pointer, IP
- Stack-Pointer, SP
- Destination-Index, DI.

Damit sind folgende konstante Beziehungen realisiert:

1. Befehlsholezyklen werden grundsätzlich im Code-Segment zusammen mit dem Instruction-Pointer durchgeführt.
2. Alle Befehle, die den Stack-Pointer benutzen (PUSH, CALL usw.), werden grundsätzlich im Stacksegment wirksam.
3. Stringbefehle, die bei der Adreßbildung den Inhalt des Destination-Index-Registers verwenden, entnehmen die Basisadresse immer dem Extrasegmentregister.

Bei der Adressierung mit Hilfe der Operanden-Register

- Base-Pointer, BP
- Base-Register, BX
- Source-Index, SI

kann durch ein Override-Präfix (1 Byte) vor dem Befehl die feste Zuordnung aufgehoben und ein beliebiges anderes Segmentregister zur Bildung der Basisadresse herangezogen werden (z. B. JMP CS: BP, Sprung auf eine Adresse im Codesegment, deren Offset durch den Inhalt von BP festgelegt ist).
Die Anwendung der Operandenregister zur Adressierung wird im Abschnitt Adressierungsmodi genauer erläutert.

2.2. *Clockgenerator 8284A*

Der Clockgenerator 8284A in der 8086-Systemarchitektur nach Bild 1.1 besitzt mit den Anschlüssen nach Bild 2.10 die Funktionseinheiten (Bild 2.11)

- Takterzeugung
- READY/WAIT-Steuerung
- RESET-Steuerung.

Effektive Adresse mit Operanden-Register	Standard-Zuordnung Segment-Register	Alternative Segmente mit Override-Prefix
Instruction-Pointer, IP	Code-Segment, CS	nein
Stack-Pointer, SP	Stack-Segment, SS	nein
Base-Pointer, BP	Stack-Segment, SS	ja
Base-Register, BX	Data-Segment, DS	ja
Source-Index, SI	Data-Segment, DS	ja
Destination-Index, DI	Extra-Segment, ES	nein

Bild 2.9 Zuordnung der Operanden-Register zu den Segmenten

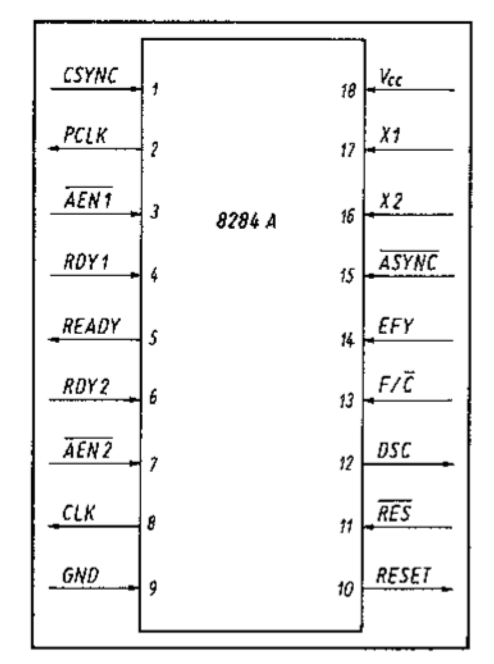


Bild 2.10 8284A-Clockgenerator

2.2.1. Takterzeugung

Die Funktion der Takterzeugung ist in Bild 2.11 dargestellt. Die Frequenz des Quarz-Oszillators wird zur Generierung von CLK einem 1:3-Teiler zugeführt. PCLK entsteht durch 1:2-Teilung aus CLK.
Zur Takterzeugung im System 8086 (Bild 2.10) werden folgende Anschlüsse des 8284A benötigt:

X1, X2 Chrystal-in (input)

An diese Pins wird der externe Quarz zur Erzeugung des Systemtaktes CLK angeschlossen. Die Quarz-Frequenz ist 3mal größer als die erforderliche Systemtakt-Frequenz.

F/C̄ Frequency/Chrystal Select (input)

Wenn dieser Eingang statisch gleich low gelegt wird, erfolgt die Realisierung des Systemtaktes durch den an X1, X2 angeschlossenen Quarz. Wenn F/C̄ gleich high ist, wird CLK durch eine an das pin EFI anzulegende externe Taktversorgung generiert.

EFI External Frequency (input)

Die externe Frequenz muß 3mal größer sein als die erforderliche Frequenz des Systemtaktes.

CSYNC Clock Synchronization (input, high-aktiv)

Der CSYNC-Eingang wird bei externer Taktversorgung mit EFI-pin benötigt und erlaubt, mehrere 8284A-Clockgeneratoren miteinander zu synchronisieren. Wenn nur ein Clockgenerator mit Quarz verwendet wird, ist CSYNC mit Masse zu verbinden.

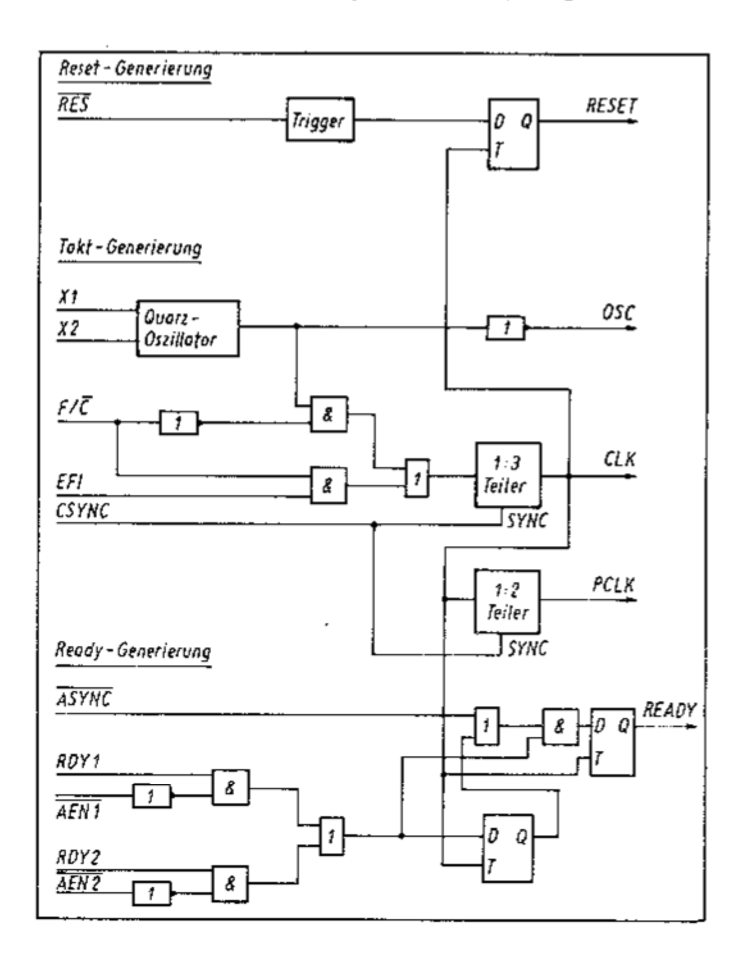
CLK Prozessor Clock (output)

Der unsymmetrische Systemtakt CLK mit der Frequenz gleich 1/3 Quarzfrequenz dient zur Taktversorgung für CPU und Buscontroller. Der H-Pegel (4,5 V) beträgt 1/3 und der L-Pegel 2/3 der Zyklusperiode (Bild 1.2).

PCLK Peripheral Clock (output)

Der Peripherietakt PCLK ist ein symmetrisches Signal mit der Frequenz gleich 1/2 CLK

Bild 2.11 8284A-Funktionsbild



und TTL-Pegel. PCLK wird für die Taktversorgung einiger Interface-Schaltkreise verwendet (z. B. 8251A-USART).

OSC Oscillator (output)

Der Takt OSC besitzt die Quarz-Frequenz mit TTL-Pegel und dient zum Treiben weiterer Clockgeneratoren an deren Eingang EFI. Eine applikative Lösung zur Realisierung des praxisrelevanten Systemtaktes 4,9152 MHz und damit der doppelten K-1520-Taktfrequenz ist in Bild 2.12 dargestellt. Die Serienwiderstände 510 Ohm an den Eingängen X1, X2 sind zur Stabilitätssicherung zwingend erforderlich.

2.2.2. READY-Synchronisation

Das System 8086 ist so ausgelegt, daß bei jedem Datentransfer eine Bestätigung von den Speichern bzw. E/A-Einheiten an die CPU erfolgt. Die Synchronisation der peripheren Bestätigung mit dem CPU-Buszyklus erfolgt im Clockgenerator 8284A durch folgende Anschlüsse (vgl. Bild 2.10, Bild 2.11):

RDY1, RDY2 Bus Ready-Transfer complete (input, high-aktiv)

Die Realisierung der Busbestätigung in Multi-Master-Systemen erfolgt an gleichwertigen Bestätigungseingängen RDY1, RDY2. Ein aktiver High-Pegel an RDY signalisiert von den peripheren Einheiten, daß die Daten empfangen bzw. gesendet wurden und der RUN-Betrieb der CPU erfolgen kann.

$\overline{AEN1}$, $\overline{AEN2}$ Access-Enable (input, low-aktiv)

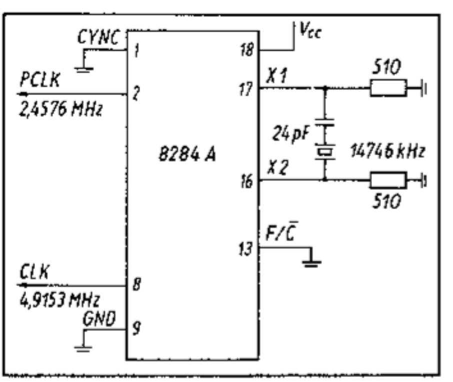
Jeder der Bus-Ready-Eingänge RDY1, RDY2 besitzt ein Torsignal $\overline{AEN1}$, $\overline{AEN2}$. Entsprechend Bild 2.11 erfolgt eine UND-Verknüpfung von RDY1&AEN1 bzw. RDY2&AEN2, die Ausgänge der UND-Gatter sind über eine ODER-Verknüpfung und Synchronisation mit CLK zum Ausgang READY geführt.

READY (output, high-aktiv)

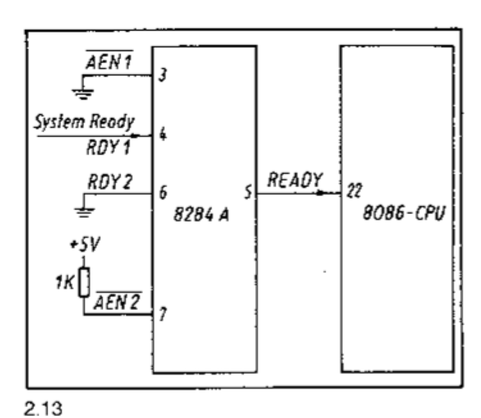
Der READY-Ausgang des Clockgenerators 8284A wird mit dem entsprechenden Eingang der 8086-CPU verbunden. Ein High-Pegel an READY-OUT-8284A gibt die synchronisierte Information für den RUN-Betrieb an die CPU weiter. READY gleich Low versetzt die CPU in einen WAIT-Zustand. Die Beschaltung des Clockgenerators 8284A zur Realisierung des RUN-Modus (System-Ready gleich High) und des WAIT-Modus (System-Ready gleich Low) über RDY1, $\overline{AEN1}$ ist in Bild 2.13 dargestellt (logische Verknüpfung von Bild 2.11 beachten!).

Wirkungsabläufe
RUN-Modus

Zur Realisierung des CPU-RUN-Modus ohne Warteschritte benötigt die CPU (vgl. Bild 2.14) ein high-aktives READY-IN-Signal mit der Set-up-Zeit von 118 ns vor der steigenden Flanke im Takt T 3. Die Übernahme der Bestätigungssignale RDY, $\overline{AEN}$ in den Clockgenerator 8284A erfolgt mit CLK am Ende von T 2. Zur internen Synchronisation müssen RDY, $\overline{AEN}$ mit den angegebenen Set-up-Zeiten aktiviert werden, um einen sicheren RUN-Betrieb zu gewährleisten. Für



2.12

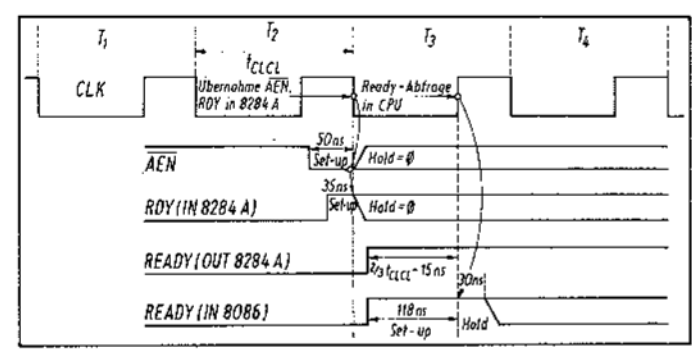


2.13

Bild 2.12 Realisierung eines 5 MHz-Systemtaktes

Bild 2.13 System-Ready mit RDY1, $\overline{AEN1}$

Bild 2.14 Zeitverläufe im CPU-RUN-Modus



2.14

einen permanenten RUN-Modus ist in Bild 2.13 RDY1 konstant gleich high zu legen.

WAIT-Modus

Die pins RDY, $\overline{AEN}$ können auch zur Realisierung eines WAIT-Modus verwendet werden, was am Beispiel der Einfügung eines WAIT-Schrittes in jeden Buszyklus erläutert werden soll (Bild 2.15).
Das Eingangssignal RDY (mit $\overline{AEN}$ gleich low) wird mit der Set-up-Zeit 35 ns vor dem Ende von T 2 inaktiv gleich low gesetzt (WAIT). Diese Information wird mit der steigenden Flanke von T 3 durch die CPU an READY abgefragt und nach T 3 ein WAIT-Zyklus Tw eingefügt. Wenn RDY vor dem Ende T 3 wieder aktiv gleich high gesetzt wird, dann folgt nach dem WAIt-Schritt Tw der Takt T 4, und der Buszyklus wird beendet.
In der applikativen Praxis ist der Generierung des RDY-Signals besondere Aufmerksamkeit zu widmen. Infolge von Verzögerungszeiten der verwendeten zusätzlichen Logikschaltkreise müssen erhebliche Toleranzgrenzen für die angegebenen Zeitbeziehungen eingehalten werden. Zur Erzeugung eines oder mehrerer WAIT-Schritte muß daher RDY bereits im Takt T 1 gleich low sein, also z. B. aus den Systemsignalen ALE oder $\overline{S0}$, $\overline{S1}$, $\overline{S2}$; AB0 ... AB19 abgeleitet werden (vgl. Bild 2.15). Eine Verknüpfung von RDY mit den Steuersignalen $\overline{MRDC}$, $\overline{MWTC}$, $\overline{AMWC}$, $\overline{IORC}$, $\overline{AIOWC}$, $\overline{INTA}$ des Buscontrollers 8288 ist zur WAIT-Auslösung nicht möglich.

2.2.3. RESET-Steuerung

Die Synchronisation des RESET-Timings erfolgt über den $\overline{RES}$-Eingang des 8284A zum RESET-Eingang der 8086-CPU (Bild 2.16). Die CPU benötigt ein mindestens 4 Takte langes high-aktives Eingangssignal RESET.

Der $\overline{RES}$-Eingang im 8284A wird über einen Schmitt-Trigger und Synchronisation mit CLK zum RESET-OUT-8284A geführt, der von der CPU abgetastet wird. Während des RESET-Timings sind die Signale des 8086-Bussystems inaktiv bzw. tristate. Die Statussignale $\overline{S0}$, $\overline{S1}$, $\overline{S2}$, die zuerst passiv-, dann tristate-Verhalten aufweisen, sind über interne pull-up-Widerstände im Buscontroller 8288 geführt, so daß entsprechend Bild 1.1 das gesamte 8086-System während des RESET-Timings einen inaktiven Zustand einnimmt. Eine applikative Lösung ohne power-on-RESET zeigt Bild 2.17.

2.3. Bus Controller 8288

Der bipolare Bus Controller 8288 (Bild 2.18) im 20-pin-DIL-Gehäuse generiert aus den CPU-Steuersignalen $\overline{S0}$, $\overline{S1}$, $\overline{S2}$ den Steuerbus des 8086-Systems (vgl. Abschnitt 1.4. und Bild 1.1).
Für die Anwendung in System-Bus-Mode in der Konfiguration von Bild 1.1 sind folgende Anschlüsse von Bedeutung:
$\overline{S0}$, $\overline{S1}$, $\overline{S2}$ Status (input)
Diese Eingangsleitungen werden mit den entsprechenden Signalen der 8086-CPU verbunden. Die Kodierung des Status ist in Abschnitt 1.1. dargestellt.

CLK Clock (input)
Systemtakt vom 8284A-Clockgenerator

IOB Input/Output Bus Mode (input)
$\overline{AEN}$ Address Enable (input)
In System-Bus-Mode nach Bild 1.1 sind die Eingänge IOB, $\overline{AEN}$ auf Masse zu legen.

CEN Command Enable (input)
Wenn der CEN-Eingang auf Low gelegt wird, dann nehmen die 8288-Command-Outputs den tristate-Zustand ein (DMA-Betrieb). Im Single-CPU-Modus ist der CEN-Eingang auf High-Potential zu legen.

Bild 2.15 Zeitverläufe im CPU-WAIT-Modus mit einem WAIT-Zyklus

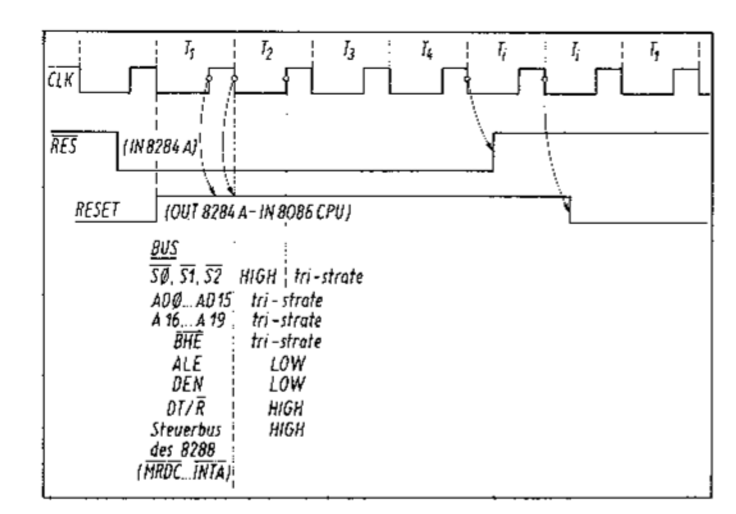


Bild 2.16 RESET-Timing

Die Command Outputs (Iol = 32 mA)
- $\overline{\text{MRDC}}$;
- $\overline{\text{MWTC}}$;
- $\overline{\text{IORC}}$;
- $\overline{\text{IOWC}}$;
- $\overline{\text{AMWC}}$;
- $\overline{\text{AIOWC}}$;
- $\overline{\text{INTA}}$

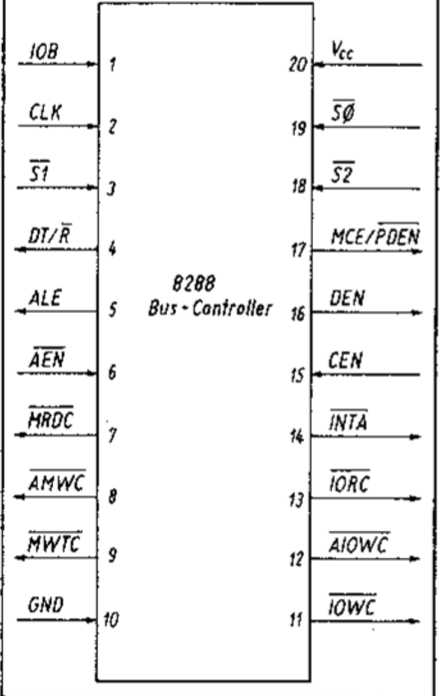
werden zum Steuerbus des Systems 8086 geführt.

Die Control Outputs (Iol = 16 mA)
- ALE;
- DEN;
- DT/$\overline{\text{R}}$

stellen die Steuersignale für die Address-Latches und die Bustreiber dar. Der zeitliche Verlauf von Command- und Control-Outputs ist in Bild 1.2 dargestellt. Die Beschaltung des Bus Controllers 8288 für eine Systemarchitektur nach Bild 1.1 ist in Bild 2.19 vorgestellt.

Literatur

/1/ MCS-86 User's Manual, Intel-Corp.
/2/ R. Rector, G. Alexy: Das 8086/8088-Buch, te-wi Verlag, 1982

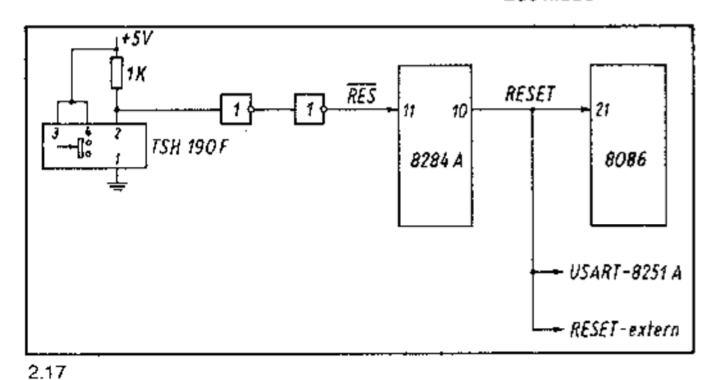


2.18

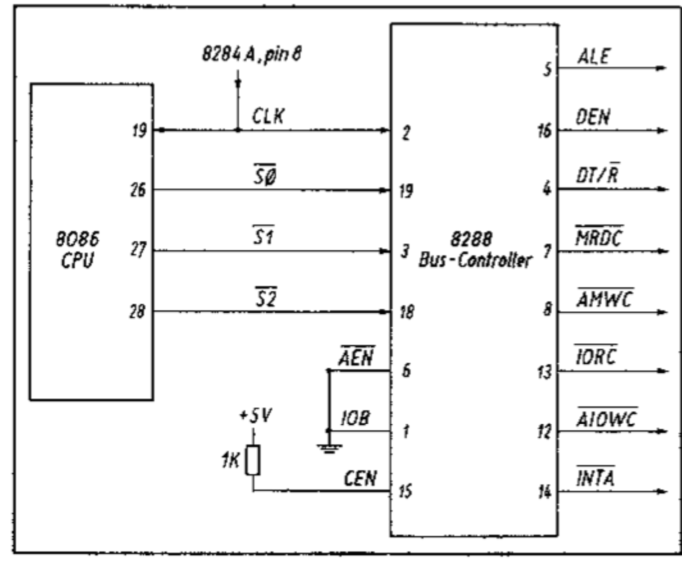
Bild 2.17 RESET-Generierung

Bild 2.18 Bus Controller 8288

Bild 2.19 Bus Controller 8288 in System Bus Mode



2.17



2.19

Eine FORTH-Systemfamilie

Dr. Michael Krapp, Jörg Richter,
Jan Schwartz,
Technische Hochschule Ilmenau, Sektion Technische und Biomedizinische Kybernetik

1. Einführung

Im Heft 6/87 der Mikroprozessortechnik wurde die ungewöhnliche Konzeption von FORTH bereits übersichtsmäßig dargestellt /1/; das gleiche Heft beinhaltet außerdem Beiträge zu speziellen Aspekten von FORTH-Implementierungen und -Anwendungen /2/, /3/. Diese und weitere Veröffentlichungen im In- und Ausland sind allgemein verfügbar und erübrigen eine nochmalige Darstellung der FORTH-Philosophie an dieser Stelle /4/ ... /10/. Seit 1982 wird an der Sektion Technische und Biomedizinische Kybernetik (TBK) der TH Ilmenau FORTH als Basis einer Programmierumgebung für Steuerungssysteme in Industrierobotern und flexiblen Fertigungsanlagen genutzt /11/ ... /15/. Dabei wurden sehr gute Erfahrungen hinsichtlich Programmierproduktivität und -qualität gemacht. Als spezifische Vorteile erwiesen sich folgende Eigenschaften, welche (neben den üblichen, aus anderen Hochsprachen bekannten Konstruktionen) FORTH auszeichnen:

- Die implizite Parameterübergabe zwischen FORTH-Worten über den Datenstapel ermöglicht eine inkrementelle Compilation, wodurch nach jedem Compilationsschritt schon während der Programmentwicklung interaktive Tests zwecks Semantik-Kontrolle eingeschoben werden können.
- Die einheitliche Strukturierung und Verwaltung von Datenobjekten und Programmen des Grundsystems sowie aller Anwendererweiterungen in Form von FORTH-Worten läßt Betriebs- und Anwendersystem, Kommando- und Programmiersprache ineinander übergehen.
- Die Kombination von Compilation (als Teil der Quelltextinterpretation) in einen speziellen Zwischencode (threaded code) und Interpretation dieses Zwischencodes (Compilatinterpretation) führt zu schneller Programmabarbeitung bei geringem Speicherplatzbedarf.
- FORTH ist transparent, da das Grundsystem selbst zu ca. 80 % in FORTH geschrieben ist und sich als Quell-Liste oder über einen Recompiler selbst dokumentiert. Der FORTH-Zwischencode erlaubt eine fast eindeutige Rückübersetzung in den Quellcode.
- Das FORTH-Grundsystem führt bei Compilation nur die notwendigsten Syntaxkontrollen durch, abgewiesen werden nur Aufrufe nicht existierender FORTH-Worte und nicht abgeschlossene strukturierte Steueranweisungen. Typprüfungen fehlen vollständig, die Laufzeitkontrolle beschränkt sich auf Überwachung des Datenstapel-Überlaufes in der Aufrufschleife. Damit werden dem Programmierer alle Freiheiten zur Systemoptimierung offen gehalten.

Die genannten Vorteile sind nicht unumstritten. Der erste hat die wenig beliebte UP-Notation mit impliziter Parameterübergabe zur Folge, der zweite und dritte verhindern eine Trennung in Code- und Datensegmente, der vierte verlangt eine relativ gute Kenntnis des FORTH-Mechanismus und der fünfte gar läßt alle möglichen „Tricks" zu, wie sie etwa im Bereich der Maschinenprogrammierung möglich sind. Deshalb muß man FORTH als Systemprogrammiersprache einstufen; sie entfaltet ihre Stärke nur bei guter Systemkenntnis und hoher Programmierdisziplin des Programmierers. Die fehlende Dokumentation der Parameterübergabe im Quelltext kann z. B. durch gute Kommentierung völlig kompensiert werden. Syntax- und Laufzeitkontrollen können durch saubere Programmierung und modularen Test unnötig gemacht oder bei ausdrücklicher Forderung im Sinne einer Spracherweiterung hinzugefügt werden. Auf diese Weise kann z. B. eine Fachsprache mit hoher Bedienersicherheit entwickelt werden, wobei FORTH für den Anwender als Basissystem unsichtbar bleibt.
Damit wird deutlich, daß ein direkter Vergleich zwischen konventionellen Programmiersprachen und FORTH nicht relevant ist. FORTH ist vielmehr als „Werkzeugkasten" für Fachsprachen u. ä. zu verstehen.
Im Rahmen der Untersuchungen zu FORTH entstanden an der Sektion TBK in den letzten Jahren u. a. drei nachnutzbare FORTH-Versionen für U880-Mikrorechner, welche gemeinsam eine Systemfamilie bilden und nachfolgend vorgestellt werden sollen. Alle drei Versionen – mit den Namen scpFORTH, singleFORTH, multiFORTH – sind im Grundwortschatz, welcher sich an fig-FORTH orientiert, identisch. Sie unterscheiden sich in der Nähe zur Rechnerhardware und in der Fähigkeit, Echtzeitanforderungen der Prozeßumgebung zu erfüllen sowie Parallelarbeit zu simulieren. Gegenüber fig-FORTH bestehen die in Tafel 1 dargestellten Abweichungen. Alle Änderungen dienen entweder der Beschleunigung der Programmabarbeitung oder der Vereinfachung des Mensch-Maschine-Dialogs. So bringt z. B. die Umstellung einiger Worte von pfa auf cfa Laufzeitgewinne. Die Fehlermeldung von (NUMBER) und NUMBER wird zur Vermeidung von unbedingten Programmabbrüchen in die Hand des Programmierers gelegt, n1 ist die aktuelle Zahlenbasis. Demgegenüber wird in ERROR eine automatische Anzeige des Fehlerortes organisiert. ?KEY legt im Unterschied zu ?TERMINAL den Tastencode auf den Datenstapel, fehlende Tastenbetätigung ergibt n = 0.
CREATE gestattet in der modifizierten Variante in Verbindung mit ALLOT die unmittelbare Definition von Datenfeldern; die CREATE benutzenden Worte wie CONSTANT usw. sind intern entsprechend geändert worden, so daß für den Nutzer kein Unterschied zu fig-FORTH entsteht. VLIST läßt sich mit ^C abbrechen, jede andere Taste unterbricht nur die Wörterbuchausgabe bzw. setzt diese fort. Mit Hilfe von Tafel 1 lassen sich fig-FORTH-Programme leicht in das hier verwendete Format konvertieren.
Um Anwendungen mit 32-Bit-Integerzahlen zu unterstützen, wurde das Grundsystem um eine entsprechende Wortgruppe erweitert. Deren Syntax ergibt sich aus der 16-Bit-Arithmetik durch vorangestelltes „D" (DDUP, DDROP, usw.). Die Terminaleingabe von 32-Bit-Zahlen wird durch einen abschließenden Punkt gekennzeichnet.

Dr. Michael Krapp wurde als Autor bereits in MP 1/88, S. 11, vorgestellt.

Dipl.-Ing. Jörg Richter (29) studierte von 1980 bis 1984 an der Technischen Hochschule Ilmenau, Sektion Technische und Biomedizinische Kybernetik. Anschließend folgte ein Forschungsstudium bis 1987. Die Forschungsarbeiten betrafen Multitask-/Echtzeitbetriebssysteme und Multiprozessorsysteme. Seit September 1987 ist Jörg Richter im Kombinat Carl Zeiss JENA als Entwicklungsingenieur tätig.

Dipl.-Ing. Jan Schwartz (32) studierte von 1977 bis 1982 an der Technischen Hochschule Ilmenau; Diplomabschluß auf dem Gebiet Technische Kybernetik und Automatisierungstechnik. Seitdem arbeitet er im VEB Robotron-Rationalisierung Weimar an Software-Entwicklungen für Montage-Industrieroboter. Seit 1984 beschäftigt er sich im Auftrag seines Betriebes an der TH Ilmenau im Wissenschaftsbereich Technische Informatik der Sektion TBK mit Betriebssystemen und Fachsprachen für flexible Automatisierungstechnik.

2. scpFORTH

Die Version scpFORTH ist als COM-Datei (SCPFORTH.COM) unter CP/M-kompatiblen Betriebssystemen lauffähig (z. B. SCP) und ist somit an jedem Arbeitsplatzrechner verfügbar. Die FORTH-I/O-Schnittstelle ist vollständig an BDOS angeschlossen. Über das FORTH-Wort BDOS kann der Nutzer selbst diese Schnittstelle erreichen. Die Arbeit mit Screens (1-KByte-Quelltextdateien) erfolgt fig-FORTH-typisch, die Screens werden aber zusätzlich von FORTH und SCP in sogenannten Screen-Dateien vom Typ SCR verwaltet.
Zum Umschalten auf andere oder zum Erzeugen neuer Screen-Dateien wird das Wort USING: verwendet, welchem der gewünschte Dateiname folgt. Innerhalb jeder Screen-Datei kann wie in fig-FORTH üblich mit einzelnen Screens gearbeitet werden; deren Numerierung beginnt also jeweils mit 0. Das Wort ?USING informiert den Benutzer über die aktuell eingeschaltete Datei. Deren Größe kann mit FILE+ bzw. FILE– in Screen-Portionen verändert werden. Die Systemvariable MAXSCR enthält die aktuell größte Screen-Nr. SCR-Dateien können mit SETRO schreibgeschützt werden, SETWR hebt diesen Schreibschutz auf. Mit BYE kann man in das SCP-Wirtsbetriebssystem zurückkehren.
Tafel 2 gibt einen Überblick über den scpFORTH-spezifischen Wortvorrat, welcher durch den Nutzer bei Bedarf erweitert werden kann. Zu beachten sind die SCP-bedingten Änderungen von ABORT und MESSAGE gegenüber fig-FORTH bzw. singleFORTH und multiFORTH.
Mit scpFORTH sind Programmentwicklungen auf FORTH-Niveau ohne Einschränkungen möglich. Problematisch sind komplette Systementwicklungen für Steuerungen, da SCP die Speicheraufteilung festlegt, welche nicht mit den gerätespezifischen Forderungen übereinstimmen muß. Nicht sinnvoll sind unter SCP außerdem Entwicklung und Test von Echtzeitprogrammen in Interruptumgebung, da SCP nicht wiedereintrittsfähig (reentrant) ist.

3. singleFORTH

Mit singleFORTH liegt ein Sytem vor, welches durch Ergänzung um gerätespezifische Treiber zu einem eigenständigen Betriebssystem wird (stand-alone-system). Es ist im Grundwortschatz kompatibel zu scpFORTH, die I/O-Worte für Terminal und Floppy entsprechen fig-FORTH, realisieren also einfache Handlerfunktionen (KEY, EMIT, R/W).
Ausgehend von den ASM-Quellen oder Objektdateien kann der Systemprogrammierer also ein auf die speziellen Belange zugeschnittenes Gerätebetriebssystem generieren. Dabei kann er durch Entkopplung der Treiberaufrufe, durch Wiedereintrittsfähigkeit der Treiber, durch Einrichtung mehrerer USER-Variablenbereiche und durch sinnvolle Positionierung der Interrupttabelle optimale Voraussetzungen für einen Echtzeitbetrieb in Interruptumgebung schaffen. Dabei sind im Einzelfall nicht alle genannten Maßnahmen notwendig; für Anwendungen in Robotersteuerungen des VEB Robotron-Rationalisierung Weimar liegt z.B. eine konkrete Implementierung für das Gerät K8915 des VEB Robotron-Elektronik Zella-Mehlis vor. Der Anwender ist mit einem solchen System in der Lage, im einfachsten Falle Interruptroutinen in Form von Maschinencode (INT-primary) in das Wörterbuch zu integrieren. Die Interrupttabelle zeigt dann sinnvollerweise auf die pfa solcher FORTH-Worte, welche natürlich nicht wie üblich als FORTH-Worte gestartet werden dürfen. Zu retten sind i.a. alle Vordergrundregister, weshalb die Hintergrundregister im Grundsystem freigehalten wurden. Zur interaktiven Initialisierung der Interrupttabellen und -quellen kann sich der Nutzer entsprechende FORTH-Worte schaffen.
Prinzipiell besteht auch die Möglichkeit, Interruptroutinen in Form von FORTH-Code (INT-secondary) zu schreiben. Die Interrupttabelle muß dann zwar immer noch auf ein INT-pri-

Tafel 1 Abweichungen von fig-FORTH (x ist n oder d)

fig-FORTH		scp/single/multiFORTH	
LATEST	(→ nfa)	ENTRY	(→ nfa)
– TRAILING	(adr n1 → adr n2)	TRAILING	(adr n1 → adr n2)
TASK	(→)	NOOP	(→)
(FIND)	(adr1 adr2 → pfa b tf)	(FIND)	(adr1 adr2 → cfa b tf)
	(adr1 adr2 → ff)		(adr1 adr2 → ff)
	(→ pfa)		(→ cfa)
NFA	(pfa → nfa)	NFA	(cfa → nfa)
(NUMBER)	(d1 adr1 → d2 adr2)	(NUMBER)	(adr n1 → x tf)
			(adr n1 → ff)
NUMBER	(adr → d)	NUMBER	(adr → x tf)
			(adr → ff)
ERROR	(n → in blk)	ERROR	(n →)
– FIND	(→ pfa b tf)	FIND	(→ cfa b tf)
	(→ ff)		(→ ff)
?TERMINAL	(→ f)	?KEY	(→ n)
CREATE	(smudge-bit = 1)	CREATE	(smudge-bit = 0)
VLIST	(abbrechbar)	VLIST	(ab- und unterbrechbar)

Tafel 2 scpFORTH-spezifischer Wortvorrat

Aufruf	Datenstapel	Bedeutung
USING: name	(b1 n →)	schließt alte, eröffnet neue Datei name.SCR
?USING	(→)	zeigt aktuelle SCR-Datei an
FCB	(→ adr)	Anfangsadresse des File-Control-Blocks
FOPEN	(→)	eröffnet FCB-Datei
FCLOSE	(→)	schließt FCB-Datei
SETRO name	(→)	setzt Schreibschutz für Datei name.SCR
SETWR name	(→)	löscht Schreibschutz für Datei name.SCR
FILE+	(scr →)	erweitert FCB-Datei bis scr
FILE–	(scr →)	kürzt FCB-Datei bis scr
MAXSCR	(→ adr)	Variable für maximale scr der FCB-Datei
BDOS	(b1 n → b2)	ruft BDOS-Funktion b1 mit n als DE-Parameter auf und hinterläßt b2 als Rückgabewert
BYE	(→)	kehrt in SCP zurück
ABORT	(→)	eröffnet zusätzlich die FCB-Datei und zeigt diese an
MESSAGE	(n →)	bei ⟨WARNING⟩ = 1 Textausgabe von Zeile n relativ zu Screen 4 der Datei FORTH.SCR des aktuellen Laufwerkes; existiert diese Datei nicht, wird mit ⟨WARNING⟩ = 0 gearbeitet

Tafel 3 Zusätzliche Dateiverwaltungsworte in singleFORTH

Aufruf	Datenstapel	Bedeutung
ERA name	(→)	löscht Datei name.SCR
SCP-DIR	(→)	Ausgabe des SCP-Directories
NEW	(→)	Umstellung von singleFORTH auf SCR-Dateiverwaltung (nach Laden der entsprechenden SCR-Dateiverwaltung)
OLD	(→)	Abschaltung der SCR-Dateiverwaltung
FLOAD name.typ	(→)	Laden der Datei name.typ in interaktiv vorzugebenden Speicherbereich
FSAVE name.typ	(→)	Auslagern des interaktiv vorzugebenden Speicherbereiches als Datei name.typ

Tafel 4 Synchronisation und Kommunikation in multiFORTH

Aufruf	Datenstapel	Bedeutung
SEMAPHORE sem-name	(→)	Definition Semaphor
EVENT ev-name	(f →)	Definition Event mit Initialwert, tf: Event eingetroffen, ff: Event nicht eingetroffen
TMESSAGE mes-name	(n →)	Definition Message, n: Message-Länge, FIFO-Organisation
sem-name	(→ sem-adr)	Semaphor-Aufruf
ev-name	(→ ev-adr)	Event-Aufruf
mes-name	(→ mes-adr)	Message-Aufruf
INIT-SEM	(sem-adr →)	Semaphor-Initialisierung „frei"
INIT-EV	(f ev-adr →)	Event-Initialisierung auf f
INIT-MES	(n mes-adr →)	Message-Initialisierung auf Länge n
RELEASE	(sem-adr →)	Semaphor-Freigabe
SIGNAL	(ev-adr →)	Event-Auslösung
SEND	(n mes-adr →)	Message-Sendung Datum n
REQUEST	(sem-adr →)	Semaphor-Anforderung
WAIT	(ev-adr →)	Event-Anforderung
RECEIVE	(mes-adr →)	Message-Anforderung
TEST-SEM	(sem-adr → f)	Semaphor-Test, ff: besetzt, tf: frei
TEST-EV	(ev-adr → f)	Event-Test, ff: nicht eingetroffen, tf: eingetroffen
TEST-MES	(mes-adr →)	Message-Test

Tafel 5 Steuerung und Zeitverwaltung in multiFORTH

Aufruf	Datenstapel	Bedeutung
PAR	(→)	Kernaktivierung
START task-name	(b →)	Start benannter Task mit Priorität b vom Terminal aus
STARTDS	(b task-cfa →)	Start benannter Task mit Priorität b vom Programm aus
PRIORITY	(b →)	setzt laufende Task auf Priorität b
SET-PRIORITY task-name	(b →)	setzt benannte Task vom Terminal aus auf Priorität b
SET-PRIORITYDS	(b task-cfa →)	setzt benannte Task vom Programm aus auf Priorität b
SCHEDULE	(→)	laufende Task gibt Prozessor an nächste ready-task gleicher Priorität ab
SUSP task-name	(→)	benannte Task wird vom Terminal aus suspendiert
SUSPDS	(task-cfa →)	benannte task wird vom Programm aus suspendiert
SUSP-ALL	(→)	außer Operatortask werden alle Tasks suspendiert
CONT task-name	(→)	benannte, suspendierte Task wird vom Terminal aus fortgesetzt
CONTDS	(task-cfa →)	benannte, suspendierte Task wird vom Programm aus fortgesetzt
CONT-ALL	(→)	alle Tasks werden fortgesetzt
KILL task-name	(→)	benannte Task wird vom Terminal aus terminiert
KILLDS	(task-cfa →)	benannte Task wird vom Programm aus terminiert
SET-TIME	(jmtsms →)	Systemuhr setzen auf Jahr, Monat, Tag, Stunde, Minute, Sekunde
?TIME	(→ jmtsms)	Abfrage Systemuhr
TIME	(→ adr n)	Abfrage Systemuhr in TYPE-Form
SLEEP	(n →)	laufende Task schläft für n Zeiteinheiten

mary zeigen, welches seinerseits jedoch nur alle Register rettet, den Returnstack (Register IX) für die Interruptroutine einstellt, die Adresse der Maschinenroutine next (im inneren Interpreter) in das Register IY lädt, die cfa des INT-secondaries auf den Datenstack legt und nicht mit NEXT, sondern mit einem Einsprung in EXECUTE endet. Das INT-secondary selbst kann bei Beschränkung auf den Datenstapel zur Parameterübergabe leicht wiedereintrittsfähig gestaltet werden. Statt des normalen Semikolon-Abschlusses dieses INT-secondaries muß dann zum Terminieren ein RETI-primary aufgerufen werden. Dieses führt nach Restauration aller Register den Maschinenbefehl RETI aus, womit dieses RETI-primary selbst und die Interrupt-Behandlung insgesamt abgeschlossen wird. INT-secondaries dürfen ebenfalls nicht wie normale FORTH-Worte gestartet werden! Auch für diese FORTH-Interruptroutine kann sich der Anwender Hilfswörter zur interaktiven Installation schaffen, so ist sogar u. U. die Definition eines Datentyps für INT-primaries sinnvoll, wenn mehrere Interruptquellen behandelt werden müssen.

Die Dateiarbeit von singleFORTH beschränkt sich auf die Screen-Arbeit von fig-FORTH. Dies ist für Steuerrechner i. a. ausreichend. Um während der Programmentwicklung kompatibel zu scpFORTH zu sein, existiert jedoch eine nachladbare SCP-Dateiverwaltung für singleFORTH, welche alle Worte einschließlich MAXSCR aus Tafel 2 realisiert, BDOS also in wesentlichen Teilen in FORTH emuliert /16/. Tafel 3 zeigt zusätzlich nachladbare Dateiverwaltungsworte, welche in scpFORTH wegen des dort verfügbaren SCP-Wirtssystems nicht notwendig sind.

4. multiFORTH

Obwohl singleFORTH schon gute Voraussetzungen zur quasiparallelen Abarbeitung von Steueralgorithmen aufweist, muß der Anwender alle notwendigen Synchronisationen, Kommunikationen und Prioritäten selbst konzipieren und implementieren. In größeren Systemen wird er damit überfordert; Effizienz, Zuverlässigkeit, Flexibilität und Transparenz sind nicht mehr gewährleistet.

Deshalb wurde auf der Basis von single-FORTH durch Unterlegung eines in Assembler geschriebenen FORTH-Multitaskkernes (FMK) das System multiFORTH entwickelt. In einer ersten Variante /14/ unterstützt dieser Kern wie polyFORTH/17/ nur eine zeitzyklische Taskumschaltung (round-robin-scheduling). Die aktuelle Variante von multi-FORTH /15/ realisiert eine verdrängende, prioritätsgesteuerte Taskumschaltung (preemptive-scheduling) und ist damit für echtzeitfähige Multitaskarbeit geeignet. Innerhalb einer Gruppe gleichpriorisierter Tasks kann durch explizites Taskumschalten (SCHEDULE, WAIT) das round-robin-Verhalten simuliert werden, welches vor allem für die interaktive Entwicklung von Tasks günstig ist, da hierbei die Operatortask automatisch und zyklisch aktiviert wird. Tasks unterscheiden sich formal bei ihrer Programmierung nicht von normalen FORTH-primaries oder -secondaries. Lediglich durch ein beim Start vorangestelltes Wort START (vom Terminal aus) bzw. STARTDS (vom Programm aus) wird ein primary oder secondary in den Task-

Tafel 6 Hilfsfunktionen

Aufruf	Datenstapel	Bedeutung
HELP	(→)	gibt alle Hilfsfunktionen aus
DIR	(scr →)	gibt ab Screen scr alle Kopfzeilen aus
MEM	(adr n →)	gibt ab adr n Zeilen Speicherinhalt aus
EDIT	(scr →)	Quelltexteditor für Screen scr
COPY	(scr1 scr2 →)	kopiert Screen scr1 in Screen scr2
SFCOPY name	(scr1 scr2 →)	kopiert Screen scr1 aus Datei name.SCR in aktuelle SCR-Datei
OVERLAY	(→)	interaktive Overlay-Technik für single/multiFORTH
RECOMP name	(→)	Recompilation des Wortes name
DISAS name	(→)	Disassemblierung des Wortes name
STEP name	(→)	Schritt-Test des Wortes name
BRK name	(→)	setzt Haltepunkt in Wort name
REPi	(→ adr)	Repetitionsvariable für i-ten Haltepunkt
SYSMAKE	(→)	Compilatauslagerung auf Diskette
SYSNAME	(→)	gibt aktuellen Compilatnamen aus
TIME-JOB	(→)	gibt Systemzeit aus
LOOK-TASKS.	(→)	gibt alle existierenden Tasks mit Zustand und Priorität aus
TCB. name	(→)	gibt TCB der benannten Task aus
SEML. name	(→)	gibt die durch benannte Tasks besetzten Semaphore aus
KRAM.	(→)	gibt symbolische Kern-RAM-Werte aus
READY.	(→)	gibt READY-Liste aus
SLEEP.	(→)	gibt SLEEP-Liste aus
SUSP.	(→)	gibt SUSPEND-Liste aus
SEM.	(sem.adr →)	gibt symbolische Datenstruktur des benannten Semaphors aus

Status erhoben; der ";"- oder "NEXT"-Abschluß führt dann zum Terminieren der Tasks. Damit ist die FORTH-Philosophie der interaktiven Programmerstellung und -testung auf multiFORTH übertragbar. Der charakteristische Datenstapel unterstützt auch hier auf natürliche Weise den Programmierer bei der Erstellung wiedereintrittsfähiger Programme.

Zusätzlich zu singleFORTH verfügt der Nutzer von multiFORTH über die Datentypen SEMAPHORE, MESSAGE und EVENT, mit deren Hilfe er Ressourcen vor Mehrfachzugriff schützen, Nachrichten zwischen Tasks austauschen sowie Tasks durch interne oder externe Ereignisse synchronisieren kann. Tafel 4 zeigt die entsprechenden Wortgruppen. Jeder Task ist beim Start eine Priorität (0 < = p < = 255) zuzuordnen, das Grundsystem meldet sich mit der Operatortask, welche die initiale Priorität 0 besitzt. Prioritäten können jederzeit vom Terminal (PRIORITY, SET-PRIORITY) oder von anderen Tasks aus (SET-PRIORITYDS) geändert werden. Außerdem stehen analog Befehle zur zeitweiligen Suspendierung (SUSP, ...) bzw. zum Fortsetzen (CONT, ...) oder zum Terminieren (KILL, ...) einer Task zur Verfügung. Tafel 5 faßt diese Steuerworte zusammen. PAR initialisiert und startet die gesamte Multitaskarbeit, die Echtzeituhr kann mit den TIME-Worten verwaltet werden. SLEEP setzt die aufrufende Task für n Zeiteinheiten (generierungsabhängig) in einen sleep-Zustand.

Nach dem Start wird jeder Task ein RAM-Bereich (Task-Segment) zugeordnet, welcher bis zur Zuordnung oder nach dem Terminieren der entsprechenden Task in einer AVAIL-Liste verwaltet wird (dynamische Taskverwaltung). Task-Segmente können nach Zahl und Größe vor PAR-Ausführung vom Anwender auf FORTH-Niveau generiert werden, initial sind Segmente für 8 Tasks eingestellt. Alle Tasks arbeiten über dem globalen Wörterbuch, welches nur von der Operatortask aus erweitert werden kann. Jede Task verfügt über lokale Daten- und Rückkehradreß-Stapel sowie über einen ebensolchen USER-Variablenbereich (dessen Belegung wird bei START „vererbt") und einen Task-Control-Block (TCB). Dieser TCB enthält die Informationen zur Priorität, zum Taskzustand (run, ready, wait, sleep, suspend), die Adresse, über welche die Tasks beim Einordnen in die Ready-, Suspend-, Semaphor-, Message- bzw. Eventwarteliste verkettet werden, sowie für den Zustand sleep die verbleibende Wartezeit.

Der Kern selbst wird von FORTH aus über spezielle primaries (Kerninterface) durch Kernrufe erreicht, welche dem Systemprogrammierer zur Verfügung stehen, um eventuell auch neue Multitaskworte zu definieren /18/. So kann für externe Ereignisse (Interrupts) z. B. ein spezieller Datentyp EX-EVENT eingeführt werden, dessen Wartetasks nicht durch SIGNAL, sondern durch eine Interruptroutine mit dem entsprechenden Kernaufruf frei gesetzt werden. Diese Interruptroutine ist dann auch Element des Kerninterface.

Die gesamte Taskverwaltung ist auf Listenverarbeitung aufgebaut, damit wird hohe Flexibilität, allerdings nicht maximale Reaktionszeit erreicht. Das Kernprogramm ist modular strukturiert, so daß auf Assemblerniveau die Möglichkeit der Kerngenerierung besteht. Ein Kern-RAM beinhaltet alle relevanten Kerndaten. Unter Nutzung dieser Daten kann sich der geübte Programmierer weitreichende Hilfswörter zur Zustandsanalyse aufbauen, welche bis zum Multitask-Debugger erweitert werden können. Der komplette Kern FMK ist einschließlich seines RAM in das FORTH-Wörterbuch integriert (ca. 4 KByte), was bei Aufruf von VLIST allerdings nicht sichtbar wird.

5. Hilfsprogramme

Obwohl FORTH selbst schon eine komplette Programmierumgebung darstellt, sind zur effizienten Arbeit meist zusätzliche Hilfsprogramme notwendig. Es ist üblich, daß sich jeder FORTH-Programmierer für seine Zwecke spezielle Werkzeuge schafft, wobei diese oft einen sehr individuellen und temporären Charakter haben. Dies ist in anderen Systemen nicht üblich und wird in FORTH neben der guten Interaktivität durch die Offenlegung aller Datenstrukturen unterstützt. So ist es oft einfacher, sich selbst eine Hilfsfunktion zu schaffen, als die Bedienung eines komple-

xen Service-Programmes zu erlernen. Einige allgemein interessierende Beispiele aus der Arbeit an der Sektion TBK sind in Tafel 6 zusammengefaßt.
Die ersten Worte bis COPY beschreiben einen Teil der üblichen Hilfsfunktionen. OVERLAY gestattet dem Programmierer, bestimmte Teile des Wörterbuches durch neue Compilate zu überschreiben, was vor allem für temporäre Hilfsfunktionen sinnvoll ist. Das Wort Overlay ist selbst interaktiv, aus seinen Bestandteilen läßt sich aber auch eine programmgesteuerte Overlay-Technik zur dynamischen Speicherplatzverwaltung aufbauen /16/. RECOMP gestattet die Recompilation des threaded-code in FORTH-Quelltext. Um alle Sprungstrukturen eindeutig rekonstruieren zu können, wurden die internen Sprunganweisungen (BRANCH, OBRANCH) aus fig-FORTH in der vorgestellten FORTH-Familie durch differenzierte Worte (z. B. (IF), (ELSE), ...) ersetzt /14/. DISAS gestattet analog zu RECOMP die Disassemblierung von primaries, wobei die SYPS-K1520-Mnemonik verwendet wird.
Die Worte STEP, BRK und REPi kennzeichnen einen ganzen Debugger-Komplex. Sowohl im Schritt-Test als auch nach Erreichen von Haltepunkten werden die obersten Werte des Daten- und Return-Stapels ausgegeben; interessieren weitere Zustandsgrößen, so sind diese dem Anwender über einen rekursiven Aufruf von FORTH (d. h. genauer: von QUIT) in bekannter Weise zur Anzeige und Veränderung verfügbar. Die Tatsache, daß man sich im Rekursions-FORTH befindet, wird lediglich durch das modifizierte Prompt-Zeichen * OK sichtbar. Nach Austritt aus dem Debugger wird wieder in das ursprüngliche FORTH zurückgekehrt.
SYSMAKE ist das Finalwort eines Komplexes, mit dem ausgetestete Anwenderprogramme unter frei wählbaren Namen in compilierter Form auf Diskette ausgelagert werden können /15/. Die entsprechenden Dateien sind SCP-kompatibel vom Typ SYF. Sie enthalten Verkettungsinformationen zu dem Basissystem, auf welches sie jeweils compiliert wurden (dies kann das FORTH-Grundsystem oder eine bereits erweiterte SYF-Datei sein) und Kaltstartinformationen. Über einen speziellen Anfangslader, welcher auf das SCP-Directory zugreift, kann man dann das gewünschte Anwendersystem als „Blatt" oder „Knoten" aus einem „Dateibaum" menügesteuert auswählen. Beim Laden werden automatisch alle SYF-Vorgängerdateien und am Anfang das FORTH-Grundsystem in der richtigen Reihenfolge der ausgewählten SYF-Datei unterlegt. Der bei SYSMAKE einzugebende Dateiname kann nach dem Laden jederzeit durch SYSNAME abgefragt werden. Der im FORTH-Grundsystem noch vorhandene Nachteil langer Compilerzeiten bei großen Anwenderprogrammen entfällt mit SYSMAKE vollständig. SYSMAKE setzt in singleFORTH die geladene SCP-Dateiarbeit voraus. Nach SYSNAME folgt in Tafel 6 eine Auswahl von Hilfsfunktionen für multiFORTH. TIME-JOB organisiert z. B. die Ausgabe von Datum und Zeit auf dem Terminal. Die darauf folgenden Worte geben interessante Systemgrößen aus, wobei soweit wie möglich die Wörterbuchnamen verwendet werden. Alle gezeigten Worte existieren auch in der Variante, daß die erfragten Informationen nicht auf das Terminal ausgegeben, sondern in Form von Adressen oder Daten auf dem Datenstapel abgelegt werden, womit dann „gerechnet" werden kann. Damit ist es möglich, sich je nach Problemstellung zugeschnittene Debugger-Tasks aufzubauen, welche gegebenenfalls interaktiv die Arbeit der Nutzertasks beobachten und analysieren. Für den naturgemäß schwierigen Prozeß der Parallelprogrammierung stehen somit interessante Werkzeuge zur Systeminspektion zur Verfügung.

☒ KONTAKT ☏

Technische Hochschule Ilmenau, Sektion Technische und Biomedizinische Kybernetik, PSF 327, Ilmenau, 6300; Tel. 740, App. 647

Literatur

/1/ Vack, G.-U.: FORTH – eine außergewöhnliche Softwarekonzeption. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 163–165
/2/ Schiemann, B.: Ein fig-kompatibles FORTH für den U8000. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 165–166
/3/ Bachmann, B.: Gleitkomma in FORTH. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 166-168
/4/ Moore, Ch. H.: Evolution of FORTH. BYTE Magazine 05/08/pp. 76–92, August 1980
/5/ Brodie, L.: Starting FORTH. Prentice-Hall 1981
/6/ Zech, R.: Die Programmiersprache FORTH. Franzis-Verlag München. 1986
/7/ Brodie, L.: Thinking FORTH. Prentice-Hall 1984
/8/ Vack, G.-U.: Mikrorechnerprogrammierung in der Dialogsprache FORTH. 27. Int. Wiss. Koll. der TH Ilmenau 1982, H. 3, S. 35–40
/9/ Woitzel, E.: comFORTH – Programmierwerkzeug FORTH unter SCP. edv-aspekte, Berlin 5 (1986) 4, S. 47–52
/10/ Fengler, W.; Roth, M.: Einchip-Rechner-Schaltkreise. Mikroprozessortechnik, Berlin 1 (1987) 2, S. 37–41
/11/ Krapp, M.; Varga, G,: FORTH als Sprachkonzept für Robotersteuerungen. edv-aspekte, Berlin 4 (1985) 1, S. 11–15
/12/ Varga, G.; Krapp, M.: FORTH-Erweiterung um Multitasking und Programmentwicklungskomponenten. 30. Int. Wiss. Koll. der TH Ilmenau 1985, Band Technische Informatik, S. 235–238
/13/ Schwartz, J.: Stand und Trend bei Roboterprogrammiersprachen. edv-aspekte, Berlin 4 (1985) 1, S. 6–10
/14/ Varga, G.: Ein interaktives, multitaskfähiges Betriebssystem für Steuerungsanwendungen. Diss. A, TH Ilmenau, Sektion TBK, 1985
/15/ Richter, J.: Ein FORTH-Multitaskbetriebssystem für flexibel automatisierte Fertigungssysteme. Diss. A, TH Ilmenau, Sektion TBK, 1987
/16/ Krapp, M.; Schwartz, J.; Alsleben, H.: Handbuch zu scp/single/multiFORTH. Forschungsbericht TH Ilmenau, Sektion TBK, 1987
/17/ Rodriguez, R. M.; Stephens, C. L.: PolyFORTH – an electronics engineer's programming toll. Software Engineering Journal, July 1986, pp. 154–158
/18/ Jahn, D.: Systemprogrammierbeschreibung für multiFORTH. Diplomarbeit, TH Ilmenau, Sektion TBK, 1987

Anforderungsspezifikation und Modellbildung auf der Basis von Netzen (Teil 2)

Dr. Stephan Fensch, VEB Kombinat Trikotagen, Karl-Marx-Stadt
Jürgen Lange, Technische Universität Karl-Marx-Stadt, Sektion Automatisierungstechnik

Ablauf des Netzmodells

Im ersten Teil dieses Beitrages wurde auf der Basis von Kanal-Instanzen-Netzen bzw. Pr/T-Netzen für ein konkretes Beispiel aus der Fertigungsindustrie ein formales Modell entwickelt, dargestellt in Bild 10 (MP 10/1987, S. 299–302). Im Sinne der Anforderungsspezifikation bildet dieses Netzmodell eine gute Ausgangsbasis zur Formulierung und Realisierung der Steuerungsaufgabe. Alle Beteiligten können anhand des Modells relativ leicht selbst überprüfen, ob ihre Forderungen komplett enthalten sind. Falsche oder noch nicht vollständige Angaben können durch weitere Verfeinerungen korrigiert bzw. ergänzt werden. Außerdem kann untersucht werden, ob Forderungen, die aus unterschiedlichen Gesichtspunkten heraus aufgestellt werden, sich eventuell widersprechen.
Im angegebenen Modell ist eine Anfangsmarkierung eingetragen, die dem Systemzustand unmittelbar vor dem Start des regulären Betriebes entspricht. Dies bedeutet, daß zum Erreichen dieses Zustands ein Einfahren der Anlage durchgeführt werden muß, welches im weiteren nicht näher betrachtet werden soll.
Durch die Markierung erhält neben den Transitionen *Bearbeitung* (4fach) und *Waschvorgang* auch die Transition mit der Prozedur *transport (robot, baz, 18)* Konzession, da der Roboter frei ist, der Kanal *WAK DWA* nicht belegt ist und an jedem Bearbeitungszentrum ein Teil auf seine Abholung wartet.
Da das Modell nichts über die Zugriffsart aussagt, wird aus dem Kanal *BAZ DWA* zufällig ein Objekt ausgewählt. Für das Teil vom Bearbeitungszentrum 1 hätte die Prozedur dann die Parameter *transport (2, 5, 18)*.
Wird vom Auftraggeber ein anderer Zugriffsmechanismus gefordert, z. B. nach Prioritäten oder nach der Reihenfolge des Fertigwer-

```
SYSTEM fms;
CONST  wak = 18;                (* phys. WAK-Position *)
TYPE   position = 1..19;        (* phys. Position im Prozess *)
       zustand = (r,b,f);       (* roh, bearbeitet, fertig   *)
       teil    = RECORD  pos:position; zu:zustand END;
       auswahl = RECORD  spa:position;
                         baz1:position;
                         baz2:position
                 END;
       str_puf  = RAM[1] OF position; (* Prädikattypen   *)
       pos_puf  = RAM[4] OF position; (* mit Zugriffsart *)
       zu_puf   = RAM[1] OF zustand;  (* und Kapazität   *)
       teilpuf3 = RAM[3] OF teil;
       teilpuf4 = RAM[4] OF teil;
       aw_puf   = RAM[3] OF auswahl;
VAR    robot, baz, spann, b1, b2:position;
BUFFER str_frei:str_puf (<2>);        (* Prädikate mit      *)
       baz_frei, sp_frei:pos_puf;     (* eventueller        *)
       wak_dwa:zu_puf;                (* Anfangsmarkierung *)
       wak_dwi:zu_puf   (<b>);
       sp_ab:teilpuf3;
       baz_dwa:teilpuf4 (<5,b> +<11,b> +<12,b> +<17,b>);
       baz_dwi:teilpuf4 (<5,r> +<11,r> +<12,r> +<17,r>);
       sp_wahl:aw_puf (<2,5,12> +<4,5,12> +<6,11,17>);
  (* Formale Definitionen von Aktionsaufrufen: *)
SEQUENCE  transport (IN pos,quelle,ziel:position);
SEQUENCE  spannen   (IN pos:position   OUT b1,b2:position);
SEQUENCE  fertigen  (IN baz:position);
```

Bild 11 Deklarationsteil des Beispiels

Bild 12 Verzögerungszellen von Transitionen als Simulationsdatenbasis

```
TRANS_BW TOKEN 4 SEC        (* Markenattribut(Token) * 4 sec *)
TRANS_SB TOKEN 4 SEC        (* Markenattribut(Token) * 4 sec *)
TRANS_WS TOKEN 4 SEC        (* Markenattribut(Token) * 4 sec *)
SPANNE RANDOM NOR 6 2 MIN(* Normalverteilt (mw=6,s=2 min) *)
BEARB TOKEN 1 MIN           (* Markenattribut(Token) * 1 min *)
DREH_B FIX 40 SEC           (* 40 sec fest *)
DREH_W FIX 40 SEC           (* 40 sec fest *)
WASCH FIX 2 MIN             (* 2 min fest *)
```

```
  0:00:00 [WASCH      ] Begin (* Beginn des Waschvorgangs *)
From (WAK_DWI   ) 1 Object: <B>
  0:00:00 [BEARB      ] Begin (* Bearbeitungszentrum 4 *)
From (BAZ_DWI   ) 1 Object: <17,R>
  0:00:00 [BEARB      ] Begin (* Bearbeitungszentrum 3 *)
From (BAZ_DWI   ) 1 Object: <12,R>
  0:00:00 [BEARB      ] Begin (* Bearbeitungszentrum 2 *)
From (BAZ_DWI   ) 1 Object: <11,R>
  0:00:00 [BEARB      ] Begin (* Bearbeitungszentrum 1 *)
From (BAZ_DWI   ) 1 Object: <5,R>
  0:00:00 [TRANS_BW  ] Begin (* Transport BAZ -> WAK *)
From (BAZ_DWA   ) 1 Object: <17,B>
From (STR_FREI  ) 1 Object: <2>
  0:02:00 [WASCH      ] End   (* Ende des Waschvorgangs *)
To (WAK_DWI   ) 1 Object: <F>
  0:05:12 [TRANS_BW  ] End   (* Ende des Transports *)
To (WAK_DWA   ) 1 Object: <B>
To (STR_FREI  ) 1 Object: <18>
To (BAZ_FREI  ) 1 Object: <17>
  0:05:12 [TRANS_SB  ] Begin (* Transport Spannpl. -> BAZ *)
From (SP_WAHL   ) 1 Object: <6,11,17>
From (STR_FREI  ) 1 Object: <18>
From (BAZ_FREI  ) 1 Object: <17>
  0:05:12 [DREH_W     ] Begin (* Start Drehwechsler WAK *)
From (WAK_DWA   ) 1 Object: <B>
From (WAK_DWI   ) 1 Object: <F>
  0:05:52 [DREH_W     ] End   (* Ende Drehen WAK *)
To (WAK_DWI   ) 1 Object: <B>
To (WAK_DWA   ) 1 Object: <F>
  0:05:52 [WASCH      ] Begin (* Beginn des Waschvorgangs *)
From (WAK_DWI   ) 1 Object: <B>
  0:07:52 [WASCH      ] End   (* Ende des Waschvorgangs *)
To (WAK_DWI   ) 1 Object: <F>
  0:10:00 [BEARB      ] End   (* BAZ 1 ist fertig *)
To (BAZ_DWI   ) 1 Object: <5,B>
  0:11:48 [TRANS_SB  ] End   (* Ende des Transports *)
To (BAZ_DWA   ) 1 Object: <17,R>
To (STR_FREI  ) 1 Object: <17>
To (SP_FREI   ) 1 Object: <6>
  0:11:48 [TRANS_WS  ] Begin (* Transport WAK -> Spannpl. *)
From (WAK_DWA   ) 1 Object: <F>
From (STR_FREI  ) 1 Object: <17>
From (SP_FREI   ) 1 Object: <6>
   ...
   ...
```

Bild 13 Kommentierter Ausschnitt aus dem Traceprotokoll

dens der Bearbeitungszentren, so kann dies durch weiteres Verfeinern der Kanäle und Objekte in gezeigter Weise in das Modell integriert werden.

Mit dem Ende der Transportoperation *BAZ1_DWA* → *WAK_DWA* wird der Kanal *WAK_DWA* mit einem Objekt ⟨B⟩ belegt, der Roboter wird auf die Position 18 freigegeben (Kanal *STR_FREI*), und im Kanal *BAZ_FREI* wird das nun beschickungsbereite Bearbeitungszentrum 1, symbolisiert durch seine Position ⟨5⟩, eingetragen. Damit ist der Transport eines Rohteils von den Spannplatzpositionen 2 oder 4 an das Bearbeitungszentrum 1 möglich, denn auf beiden Spannplätzen ist dieses Bearbeitungszentrum angewählt (Kanal *SP_WAHL*, Objekte ⟨2, 5, 12⟩ *bzw.* ⟨4, 5, 12⟩) und damit die zusätzliche Bedingung (baz = b1) or (baz = b2) durch baz = b1 = 5 erfüllt.

Nach Ausführung des Transports wird ein Spannplatz freigegeben (Kanal *SP_FREI*), so daß, wenn der Waschvorgang abgeschlossen ist und der Drehwechsler gedreht hat, das Fertigteil von der Waschstation auf den freien Spannplatz zum Abspannen transportiert werden kann.

Auf diese Weise können mit dem geschaffenen Modell zahlreiche interessante Situationen des Systems durchgespielt werden, was allerdings ohne Rechnerunterstützung sehr uneffektiv ist. Außerdem wäre es wünschenswert, auf Basis des einmal geschaffenen Modells auch schon einige Simulationsexperimente durchführen zu können, die Aussagen über Laufzeiten, Auslastung und kritische Situationen im zukünftigen System zulassen.

Das Programmpaket POSES

Zu diesem Zweck wurde an der TU Karl-Marx-Stadt das Sprachsystem POSES (Pr/T-Netzorientiertes Simulations- und Entwurfssystem) /4/ geschaffen, mit dessen Hilfe Netzmodelle formal notiert und auf einem Personalcomputer simuliert (interpretiert) werden können.

Grundkonzepte

Die semantische Basis der Sprache bildet das „Situation-Handlungs"-Prinzip. Drei wesentliche Konzepte bilden die programmtechnische Grundlage von POSES:

- Datenflußkonzept:

Eine konsequente Umsetzung des Markenflusses aus dem Netzmodell führte zwangsläufig dazu, daß POSES als Datenflußsprache /5/ konzipiert wurde. Sie unterscheidet sich gegenüber den imperativen Programmiersprachen (FORTRAN, PL1, ALGOL, PASCAL, ...), deren Abarbeitung vom Stand eines Befehlszählers gesteuert wird, insbesondere dadurch, daß die Ausführung von Programmaktivitäten (Instanzen im Netz = *Handlungen*) von aktuellen Datenwerten (Markenverteilungen im Netz = *Situationen*) abhängt, die damit nicht nur passive Werte-Repräsentanten sind.

Die Bedeutung des Begriffs Variable differiert wesentlich von der Vorstellung, daß es sich einfach um überschreibbare benannte Speicherzellen handelt, sondern entspricht mehr der von logischen Programmiersprachen (PROLOG). Die Variable in POSES kann für die Dauer der Abarbeitung einer Transition mit Markenattributen belegt werden und ermöglicht dann Abfragen über Relationen zu oder das Erfülltsein von weiteren Markenattributen (etwa im Sinne von PROLOG). So werden z. B. die Variablen baz mit 5 bzw. b1 mit 12 belegt und dann gefragt, ob baz = b1 gilt. Ist das nicht der Fall, werden weitere Belegungen der Variablen baz und b1 versucht, was zum sogenannten Backtracking führt.

- Typkonzept:

Jede Konstante, Variable, jeder aus diesen gebildete Ausdruck und jedes Prädikat ist in POSES von einem bestimmten Typ. Dieser Typ bezeichnet die Wertemenge, der eine Konstante angehört bzw. die durch eine Variable oder einen Ausdruck angenommen werden kann, oder den Typ der Objekte, die ein Prädikat aufnehmen kann. Jeder Operator und jedes Prädikat erwarten Argumente bzw. Objekte eines bestimmten Typs. Durch diese Redundanz werden schon zur Compilationszeit Entwurfsfehler erkannt und können somit vermieden werden.

- Vertikale und horizontale Dekomposition:

a) Vertikale Dekomposition: Die Aufteilung eines Steuerungssystems in eine Zwei-Ebenen-Hierarchie (Kontroll- und Aktionsebene) gestattet, unterschiedliche Systemaspekte separat zu behandeln und die geeignetsten Mittel und Methoden für die jeweilige Ebene einzusetzen. POSES realisiert diesbezüglich das Konzept der Sprachfamilie. Die Netzbeschreibung bietet leistungsfähige Konstruktionen für die Beschreibung paralleler Strukturen der Kontrollebene. Für die Angabe der sequentiellen Aktionen (Kopplung zum physischen Prozeß) kann der Programmierer auf andere Sprachen (Assembler, PASCAL ...) zurückgreifen, die hier vorteilhafter sind.

b) Horizontale Dekomposition: Während bei der vertikalen Dekomposition das Steuerungssystem in Hierarchieebenen aufgeteilt wird, wird bei der horizontalen Dekomposition eine Modularisierung der Kontrollebene selbst vorgenommen /6/. Jeder Modul bildet in POSES eine Kapsel mit genau definierten Schnittstellen. Diese bestehen aus Prädikat-

```
State: S1  (STR_FREI  ) 1 Object  (* S1 ≙ Situationsname *)
From: 0:00:00  To: 1:03:24
Occured: 12
Time: 0:00:00 => 0%

State: S2  (STR_FREI  ) 1 Object  (* S2 ≙ Situationsname *)
From: 1:03:24  To: 2:00:27
Occured: 36
Time: 0:06:24 => 11%

State: S3  (STR_FREI  ) 1 Object  (* S3 ≙ Situationsname *)
From: 2:05:52 To: 3:05:02
Occured: 14
Time: 0:39:59 => 67%
```

Bild 14 Protokoll

paaren der Form (Pi, Po), wobei Pi der Eingang und Po der Ausgang für den Datenfluß ist. Für die Verkopplung der einzelnen Moduln untereinander wird durch eine spezielle Entwurfsstrategie die reine Baumstruktur angestrebt, die aber durch sogenannte Ressourcen-Moduln auch aufgelöst werden kann.
Ein POSES-Programm besteht aus einem oder mehreren derartigen Modulen, wobei jeder Modul aus Deklarations- und Netzteil besteht.

a) Deklarationsteil: Im Deklarationsteil sind die Konstanten, die Typbezeichner mit ihren Wertemengen, die Variablen, die Prädikate mit ihrer Anfangsmarkierung, die Koppelpaare des Moduls, die Funktionen und die Kopfzeilen der Aktionsprozeduren zu definieren, die den Datenfluß von der Kontroll- über die Aktionsebene zum physischen Prozeß und umgekehrt realisieren. Den Deklarationsteil des Beispiels zeigt Bild 11.

b) Netzteil: Im Netzteil ist die Topologie des Netzes anzugeben. Der Netzteil entsprechend Bild 10 ist in Ermangelung grafikfähiger Hardware in eine verbale Notation formal umzusetzen, die an dieser Stelle nicht weiter dargestellt werden soll. Zum aktuellen, auf CP/M-, CP/M-86- und MS-DOS-kompatiblen Betriebssystemen lauffähigen, nachnutzbaren POSES-System gehören Compiler, Modulverbinder, Interpreter für Zwischencode und ein Simulationspaket.

Simulation mit POSES

Der Simulator erlaubt neben der Austestung des Markenspiels eine komfortable Simulation des beschriebenen Prozesses. Dadurch sind bereits vor dem Entwurf der eigentlichen Steuerung (der ebenfalls mit POSES möglich ist) Aussagen über
– kausale Zusammenhänge des Prozesses
– Auslastungskoeffizienten (z. B. von Transportmitteln)
– zeitliche Abläufe
– (Schalt-)häufigkeiten, Engpässe usw.
möglich.

Der POSES-Simulator entspricht somit dem Anliegen und erfüllt einen beträchtlichen Teil des Funktionsumfanges von SIMDIS, versteht sich allerdings nicht als Konkurrenz dazu. Dafür steht als Hauptgrund, daß der Simulator nur ein Teil des durchgängigen POSES-Systems von der Modellierung mit Hilfe der Pr/T-Netze bis zur Realisierung der Steuerung ist. Mit der Analyse und formalen Modellierung des Prozesses zum Zwecke der Simulation sind bereits wesentliche Komponenten zur Steuerung dieses Prozesses in das Netzmodell integriert worden. Damit ist nach relativ geringen Änderungen im Netzmodell (Interruptstellen und Aktionsproceduraufrufe zur Verkopplung der Steuerung mit dem Prozeß) die Implementierung der Prozeßsteuerung rechnergestützt realisierbar. Bei der Simulation kann die zu simulierende Zeit in Abhängigkeit von Markenattributen berechnet werden (z. B. soll sich die Transportzeit proportional zur Entfernung zwischen Quell- und Zielposition verhalten).

Der POSES-Simulator ist vor Beginn, während und nach Ende des Simulationslaufs im Dialogbetrieb mit unterschiedlichen Hilfegraden bedienbar. Die Aufbereitung und Auswertung der Simulationsergebnisse ist in den Simulator integriert worden, läuft grundsätzlich für alle Prädikate und Transitionen intern mit und ist jederzeit abrufbar. Darüber hinausgehende Interessen, z. B. eine spezielle Markierung, die mehrere Prädikate umfaßt, können explizit formuliert werden.
Es sind 3 Hauptklassen von Funktionen realisiert worden:

● Funktionen, die den Netzablauf, der im Schritt- oder Run-Betrieb möglich ist, beeinflussen
o Zuweisung von Zeiten an Transitionen, um die hypothetische oder bekannte Dauer von diesen Transitionen zugeordneten Aktionsprozeduren (z. B. eines Transportes) dem Simulator mitzuteilen – als
– feste Zeit
– Zufallszahl (verschiedene Verteilungen möglich)
– Zeit ist eine Funktion von Markenattributen
o Ändern der aktuellen Markierung
o Ändern der Priorität von Transitionen ...

● Funktionen zur visuellen und internen Protokollführung
o Festlegen des Ausgabeumfangs von Transitionen
– ob beim Schalten einer Transition eine entsprechende Meldung ausgegeben wird
– Kantenbelegung beim Schalten
– warum schaltet Transition nicht
o Festlegen von zu protokollierenden Markierungen
o Festlegen des zu protokollierenden Moduls

● Funktionen zur Auswertung und Anzeige netz- und simulationsspezifischer Daten
o Auslisten der Prädikate
– aktuelle Markierung, Kapazität, Zugriffsstrategie
– maximale und zeitlich gemittelte Markenzahl
– prozentuale Auslastung
o Auslisten der Transitionen
– Priorität, Verzögerungsart
– Schalthäufigkeit, Schaltdauer u. a.
o Auslisten von Markierungen
– wie oft und wie lange in welchem Zeitraum ist Markierung eingetreten.

Simulation des Beispiels

Mit dem durch Editieren und Compilieren aufbereiteten Beispiel wurden einige Experimente mit der in Bild 12 gezeigten Simulationsdatenbasis durchgeführt.
Allen Transitionen wurden technologisch bedingte Abarbeitungszeiten zugeordnet, um Aussagen über das zeitliche Verhalten des Prozesses zu erhalten. Da die Transportzeit linear von der Entfernung Quell-Zielposition abhängt, kann sie aus den Markenattributen berechnet werden. Dieser Zahlenwert wird zusätzlich mit einem modifizierbaren Faktor und der Einheit versehen.
Die Darstellung der Ergebnisse erfolgt in Simulationsprotokollen, die ablauforientierte Teile (trace) und Analyseteile aufweisen. Einen Ausschnitt aus einem solchen Protokoll zum Beispielsystem zeigt Bild 13.
Interessant sei die Auslastung des Transporters. Für jeweils ca. 1 Stunde sollen die 3 Situationen
a) normaler Betrieb
b) 4fache Verfahrgeschwindigkeit des Transporters
c) wie b), aber Ausfall von 2 Bearbeitungszentren
nacheinander untersucht und analysiert werden.
Die Auslastung des Transporters entspricht der zeitlichen Markenbelegung des Kanals *STR_FREI*.
Die Simulationszeit wird auf jeweils 1 Stunde (keine Echtzeit) festgelegt, und danach werden die Resultate ausgegeben. Die erhöhte Verfahrgeschwindigkeit wird durch Ändern des Faktors in der Simulationsdatenbasis auf 1 erreicht (z. B. TRANS_WS TOKEN 1 SEC).
Der Ausfall von Bearbeitungszentren kann durch „manuelle" Entnahme von Marken aus dem Netz, z. B. 2 Marken vom Kanal BAZ_DWI, simuliert werden.
zu a) Bild 14a
Der Transporter war zwar 12mal frei, wurde aber offenbar ohne Verzug sofort wieder belegt und war somit 100%ig ausgelastet.
zu b) Bild 14b
Der Transporter war 36mal frei und ruhte 6:24 min. Die Auslastung ergibt sich zu 100 % – 11 % = 89 %.
zu c) Bild 14c
Der Transporter war 11mal frei und ruhte 39:59 min. Die Auslastung ergibt sich zu 100 % – 67 % = 33 %.
Für statistisch gesicherte Simulationsergebnisse müßte die Simulationszeit sicher höher bemessen werden. An dieser Stelle sollte nur ein kleiner Einblick in die Möglichkeiten des Simulationssystems gegeben und demonstriert werden, daß schnell und unkompliziert verschiedenste Experimente durchführbar sind. *Schluß*

Literatur

/4/ Meyer, G.; Fensch, S.; Kunke, W.; Lange, J.: Softwareentwurf in der Steuerungstechnik. In Roth (ed.): Beiträge zur Mikrocomputertechnik, Verlag Technik Berlin 1986
/5/ Myers, G. J.: Data flow machine architecture. Advances in computer architecture, chap. 22, John Wiley & sons
/6/ Wirth, N.: MODULA-2. Berichte des Instituts für Informatik, ETH Zürich, Vol. 36

REDABAS-Entwicklungshilfe

Die rasche Ablösung der vielen kleinen Datenbestände, die bisher auf Karteikarten u. a. gehalten wurden, ist effektiv nur durch den Einsatz eines Datenbanksystems (z. B. REDABAS/dBase) möglich. Für die Erfassung und Verwaltung dieser Datenbestände wurde eine allgemeine Kommandodatei entwickelt mit folgenden Funktionen:

● schnelle Eingabe von Datensätzen (Erweitern der Datei) mit den Entscheidungen:
– Korrektur des erfaßten, aber noch nicht abgespeicherten Satzes
– Abspeichern des Satzes und Erfassen des nächsten Satzes
– Verwerfen des soeben erfaßten Datensatzes

● komfortable Korrektur von Datensätzen (Verwaltung des Datenbestandes ohne weitere Hilfsmittel) mit den Unterfunktionen:
– Vor- und Rückwärtsblättern in der Datei, dabei wird je ein Satz (entspricht einer Karteikarte) auf dem Bildschirm angezeigt
– Editieren (d. h. Ändern und Korrigieren) des aktuellen Satzes
– Löschen des angezeigten Satzes
– Finden des nächsten Satzes lt. Schlüsselfeld.

Mehrere dieser Kommandodateien werden durch eine Hauptkommandodatei bedient. Außerdem ist noch ein Anschluß für den Listendruck möglich. Für die Anwendung dieser Lösung wurde ein TURBO-PASCAL-Programm dBKartei (Datenbank-Kartei) entwickelt, welches die Grundstruktur eines solchen Projektes erzeugt. Das Programm dB-Kartei ist auf BC/PC (8-Bit-Technik) unter einem CP/M-kompatiblen Betriebssystem einsetzbar und für REDABAS und dBase gleichermaßen geeignet. Bei einer Laufzeit von etwa 10 Minuten kann ein (kleines!) Datenerfassungsproblem zusammen mit REDABAS/REDAMASK oder dBase/ZIP innerhalb weniger Stunden gelöst werden.

VE Forschungsinstitut für Obst- und Gemüseverarbeitung, Nicolaistr. 5, Magdeburg, 3018; Tel. 222112 App. 49

Gaßmann

Mikrorechner-baugruppensystem

Das NANOS-Mikrorechnerbaugruppensystem ist eine Entwicklung der Ingenieurhochschule für Seefahrt Warnemünde/Wustrow und wird zentral im Betriebsteil (BT) Entwicklungs- und Produktionsstätte mikroelektronischer Rationalisierungsmittel (EPMR) des VEB Datenverarbeitungszentrum (DVZ) Rostock gefertigt.

Es bietet hard- und softwareseitig die Möglichkeit, den Schritt von sogenannten Insellösungen zu komplexen, mit den Prozessen verknüpften Rechnersystemen zu vollziehen. Mittels des NANOS-Sortimentes lassen sich die verschiedensten Ausführungen der Prozeßanbindung realisieren wie
– Einsatz als Ergänzungsmodul
– Einsatz als spezialisiertes NANOS-Prozeßterminal in Verbindung mit dem PC über ein Standard-Geräteinterface (Feldbus) oder
– Einsatz als NANOS-Terminal im lokalen Netz.

Die Anwendungsgebiete sind äußerst vielfältig, z. B.:
– Automatisierung von Laboratorien
– Automatisierung von einzelnen industriellen Prozessen ohne hohe Echtzeitforderungen
– automatische Steuerung von Sondermaschinen und Rationalisierungslösungen
– Automatisierung der Produktionsdatenerfassung
– Automatisierung der Erfassung von TUL-Prozeßdaten
– Steuerung von Meß-, Prüf- und Simulationseinrichtungen
– effektivere Auslastung hochwertiger Rechnerperipherie.

Der Systembus des NANOS-Baugruppensystems entspricht im wesentlichen der TGL 37271 und ist somit elektrisch zum K-1520-System kompatibel (Prozessorbasis U880). Durch die Orientierung auf Systemkompatibilität zu CP/M können leistungsfähige Rechnerarbeitsplätze mit NANOS-Baugruppen realisiert werden. Auf Basis eines speziellen CP/M-kompatiblen Betriebssystems können so CP/M-Originalprogramme von Büro- und Personalcomputern teilweise übernommen werden.

VEB DVZ Rostock, BT EPMR, E.-Schlesinger-Str. 37, Rostock 6, 2500; Konsultationen zu technischen und kommerziellen Fragen: Abteilung Absatz (RA) Tel. 85 11/41

Habetha

Grafiksystem zur Arbeit unter SCP

Für die in MP 2/1987, Seite 61, beschriebene Farbgrafiksteckeinheit CGD (unter UDOS arbeitend) wurde zur Ansteuerung der 256 × 256 Bildpunkte in 8 Farben mit je zwei Helligkeitsstufen ein universell einsetzbares Grafikpaket in PASCAL unter SCP entwickelt.

Das modular aufgebaute Grafikpaket besteht aus folgenden Teilen:
– Hardwarespezifische Anpassung für CGD-Steckeinheit
– Grafiksystem mit grundlegenden Grafikroutinen
– Routinen zur Textdarstellung.

Daneben existieren noch weitere Anpassungen für die Steckeinheit VIS 2A von der Akademie der Wissenschaften der DDR und für den Drucker EPSON LX-86. Das Grafiksystem beinhaltet grundlegende Grafikbefehle wie:
– Festlegen der Zeichenfarbe (COLOR)
– Zeichnen eines Punktes und einer Linie (PIXEL, LINE)
– Festlegen des Betrachtungsfensters (WINDOW) im Nutzerkoordinatensystem
– Festlegen des Darstellungsfensters (VIEW) am Bildschirm
– absolutes und relatives Zeichnen mit Clippen
– Füllen von Teilen und des gesamten Bildschirmes mit einer Farbe
– Schreiben und Einlesen eines Bildes von Diskette u. a.

Die Darstellung von Texten in verschiedenen Größen und Richtungen ist möglich, wobei wahlweise Texthintergrund ein- oder ausgeblendet werden kann.

Folgende Anwenderprogramme wurden unter Verwendung des Grafiksystems entwickelt:
– Zweidimensionale Darstellung von Funktionen und Zeitreihen
– dreidimensionale Darstellung von Funktionen mit oder ohne verdeckte Linien bei freier Wahl des Betrachtungspunktes
– Geschäftsgrafik (in Bearbeitung).

Forschungsinstitut für Textiltechnologie, Abt. PM, PSF 243, Karl-Marx-Stadt, 9010; Tel. 57090/436.

Dr. Gebhardt

Programmsystem Personalstatistik

Das Programm dient der Erfassung, Speicherung und Verarbeitung von Personaldaten, um den Arbeitsaufwand bei statistischen Analysen wesentlich zu senken. Mit diesem Programm ist es möglich, von jeder Person 43 verschiedene Merkmale zu speichern, die sich aus den Bereichen Personalien, betriebliche Kriterien, Parteien und Massenorganisationen, gesellschaftliche Funktionen, Auszeichnungen und Bildungsniveau zusammensetzen. 12 Merkmale sind durch ein Initialisierungsprogramm vom Anwender selbst modifizierbar. 51 Merkmale können zu einer logischen UND-Verknüpfung zusammengestellt werden, nach denen die Stammdatei selektiert wird.

Das Programmsystem erfüllt weitere Funktionen wie:
– Erstellen, Ändern, Löschen von Datensätzen
– Suchen in der Stammdatei nach festen und variablen Merkmalen
– Anfertigen von Erfassungsbelegen.

Die Ausgabe der Daten erfolgt wahlweise auf dem Bildschirm oder Drucker.

Je nach Diskettengröße und Formatierung ergeben sich Speicherkapazitäten für 320 bis 1600 Personen. Das Programmsystem ist in BASIC geschrieben und lauffähig unter dem Betriebssystem SCP mit einem Bildschirmformat von 80 × 24 Zeichen.

VEB Robotron-Büromaschinenwerk „Ernst Thälmann" Sömmerda, BfN, Weißenseer Str. 52, PSF 43, Sömmerda, 5230; Tel. 43257 oder 42876.

Schröder

Gravur von Leiterplattennegativfilmen

In unserem Betrieb werden die Negative zur Leiterplattenbelichtung nicht mehr über den photographischen Umweg der Druckstockzeichnung hergestellt, sondern mittels eines Zeiss-Zeichentisches DZT 90 × 120 auf Gravurfolie graviert. Mit diesem Verfahren sind Schwierigkeitsstufen bis zu V durchaus zu beherrschen. Angesteuert wird der DZT von einem CP/M-2.2-kompatiblen Bürocomputer. Wir bieten dazu ein Steuerprogramm. Die Koordinatendaten werden bei uns von einem Digitalisiergerät DG 20 geliefert. Eine nutzereigene Anpassung an verschiedene Eingabeformate ist möglich. Für die Gravur eines Filmnegativs im K-1520-Format werden etwa zwei Stunden benötigt.

BKK Bitterfeld – Stammbetrieb – Territorialbereich Halle, DB FOPRO, Abt. Prozeßautomatisierung, Eisenbahnstraße 10, Halle, 4073; Tel. 464391 (Koll. Oertel)

Dr. Hiekel

Emulatormodul für U882 unter CP/M und FORTH

Für die Bedienung des U881/U882-EMR-Entwicklungsmoduls aus dem ZFT des KEAW wurde eine Terminalsoftware, die unter CP/M-2.2-kompatiblen Betriebssystemen lauffähig ist, geschrieben.

Weiterhin wird eine Lösung zur Entwicklung von FORTH-Programmen für dem EMR angeboten. Dazu gehören das FORTH-Laufzeitsystem, eine Bibliothek von z. Z. 54 FORTH-Worten und ein auf comFORTH basierendes Crosscompilersystem. Auf der Basis der Koppelsoftware unter CP/M 2.2 ist ein effektiver Test der FORTH-Programme auf dem EMR-Entwicklungsmodul möglich. Implementiert sind z. Z. Funktionen wie Datenstackanzeige, Haltepunkte auf FORTH-Niveau mit symbolischer Eingabe.

BKK Bitterfeld – Stammbetrieb – Territorialbereich Halle, DB FOPRO, Abt. Prozeßautomatisierung, Eisenbahnstraße 10, Halle, 4073; Tel. 464391 (Koll. Oertel)

Dr. Hiekel

Driver zur Kopplung DZT 120 × 90 – PC 1715

Die Ansteuerung des Digitalzeichentisches DZT 120 × 90 RS mittels PC 1715 unter dem Betriebssystem SCPX ist nach entsprechender Installation des Betriebssystems über den Printer-Driver möglich und üblich. Das hat jedoch zwei wesentliche Nachteile: Einmal ist der Parallelbetrieb Drucker/Zeichentisch in den wenigsten Fällen sinnvoll möglich, zum anderen kann der wichtige READ-Befehl des DZT-Befehlssatzes nicht genutzt werden.

In unserem Bereich wurde von einem Jugendforscherkollektiv ein Driver entwickelt, der als Maschinenunterprogramm in BASIC, FORTRAN, REDABAS den Datenaustausch PC – DZT über die V.24-Schnittstelle realisiert. Der Read-Befehl des DZT sowie ein unabhängiger Betrieb von Drucker und Zeichentisch sind möglich.

Mit geringfügigen Änderungen kann dieser Driver auch für den BC A 5120/5130 sowie für IFSB-Schnittstellen genutzt werden.

VEB Mansfeld Kombinat „Wilhelm Pieck", Werk Kupferbergbau/Markscheiderei, Markt, Lutherstadt Eisleben, 4250; Tel. 2641

Dr. Wordelmann

Aufbau und Arbeitsweise von 16-Bit-Mikroprozessoren

von W. Grafik, Reihe Automatisierungstechnik, Bd. 226. Berlin: VEB Verlag Technik 1987

In dieser Broschüre wird ein vergleichender Überblick zur Leistungsfähigkeit dreier typischer Vertreter von 16-Bit-Mikroprozessoren (U8000, K1810WM86, MC68000) gegeben.
Nach einem allgemeinen Überblick zur Entwicklung der Mikroprozessortechnik werden im ersten Teil die Architekturen der Prozessoren vorgestellt. Dabei stehen vor allem Hardwareaspekte, wie Prozessorstruktur, Pinbelegungen, Aufbau der Registersätze, sowie die unterschiedlichen Arbeitsweisen zur Speicherverwaltung im Vordergrund.
Im zweiten Teil folgt eine ausführliche Darlegung zu den verschiedenen Adressierungsarten aller drei Prozessoren. Dieser auf die Assemblerprogrammierung ausgerichtete Teil wird durch eine Reihe von Beispielen, Bildern und Tafeln aufgelockert. Die gewählte Darstellungsweise und die vergleichenden Übersichtstafeln ermöglichen dem Leser eine schnelle Einarbeitung in grundlegende Prozessoreigenschaften.
Obwohl im Anhang die Befehlsliste zum MC68000 (dem im Inhaltsverzeichnis eine Null verloren ging) fehlt, wurde das schwierige Problem, auf dem durch die Reihe Automatisierungstechnik begrenzten Seitenumfang drei 16-Bit-Mikroprozessoren zu vergleichen, gut gelöst.

Ulrich Oefler

VEM-Handbuch Automatisierungsanlagen

von einem Autorenkollektiv
VEB Verlag Technik, Berlin, 1. Auflage, 384 S., 211 Abb., 80 Taf., 38,– M

Mit der breiten Einführung des mikrorechnergestützten Prozeßleitsystems „audatec" des VEB GRW „Wilhelm Pieck" Teltow zur Automatisierung verfahrenstechnischer Prozesse ist aus der praxisorientierten VEM-Handbuchreihe das Fachbuch „Automatisierungsanlagen" über den Volksbuchhandel verfügbar.
12 Fachleute mit langjähriger Berufserfahrung (Red: Dr.-Ing. J. Kliemann) haben ihre bei Entwicklung, Errichtung und Nutzung von Automatisierungssystemen auf Mikrorechnerbasis gewonnenen Erkenntnisse zusammengefaßt und vermitteln anschaulich in acht Kapiteln Grundwissen über die Arbeitsweise dezentral strukturierter Prozeß-Leitsysteme.

Nach kurzer **Einführung** zu den generellen Entwicklungstendenzen und allgemeinen Hinweisen zur **Gestaltung von Automatisierungsanlagen** werden verfüg- und absehbare Anlagentechnik einzeln und im Komplex beschrieben sowie Einsatzvarianten in verschiedenen Industriezweigen angeboten. Im Hauptteil **dezentrale Automatisierungsanlagen** entwickeln die Autoren ein Anlagenaufbaukonzept von der Konfiguration über die Informationsgewinnung und -übertragung bis zur Anlagensicherheit, dem schließt sich eine umfangreiche und detaillierte Beschreibung neuer Baugruppen (z. B. Basissteuereinheiten, Pulssteuerrechner, Datenbahnsteuerstation, Monitore) an. Besonders wertvoll für den Nutzer ist die ausführliche und informative Erläuterung der Software. Im weiteren findet man Hinweise zur Prozeß- und Systemkommunikation und zu verfügbarkeitserhöhenden Maßnahmen.
Fertigung, Montage und Inbetriebnahme sind in ihren Hauptetappen und bezogen auf Baugruppen, Gefäße und Anlagen erläutert; Instandhaltung, Kundendienst und Schulung von Herstellern und Anwendern schließen den Problemkreis. Eine Auswahl bereits realisierter Anwendungsbeispiele in der Chemieindustrie, der Kraftwerksautomatisierung und der Metallurgie soll gewonnene Erfahrungen einem breiten Interessentenkreis zugänglich machen.
Nachgestellt sind ein ausführliches **Verzeichnis internationaler gesetzlicher Grundlagen, Verordnungen und Standards** sowie ein umfangreiches, abschnittsbezogenes **Literaturverzeichnis**. Das Werk ist kurz und sachlich, präzis und informativ gehalten. Zahlreiche Bilder und Tafeln erleichtern zudem das Verständnis und mit der sorgfältig ausgesuchten Sammlung von Standards und Literatur folgen die Autoren einem begrüßenswerten Trend.
Der Verwendungszweck reicht damit vom Nachschlagewerk bis zum Lehrbuch und wird besonders all jenen nützlich und zu empfehlen sein, die sich mit Entwicklung, Projektierung und Betrieb o. a. Technik befassen oder artverwandte Fachrichtungen studieren oder lehren.
Sie alle würden sich aber über eine bessere Papier- und Druckqualität in der 2. Auflage freuen, da dieses Fachbuch unter den rauhen Betriebsbedingungen vor Ort in allen potentiellen Einsatzfällen sicher schnell verschleißt.

Klaus G. Amler

PASCAL: Einführung – Programmentwicklung – Strukturen

von J. Plate/P. Wittstock, 3., durchges. und erweit. Auflage.
Franzis-Verlag GmbH München 1986, 426 Seiten, 59,– DM

Das vorliegende Buch setzt praktisch keinerlei Vorkenntnisse auf dem Gebiet der Rechentechnik voraus. Es wendet sich somit an den großen Kreis der Einsteiger in die Computerprogrammierung und bietet des weiteren dem Praktiker, der auf PASCAL umsteigen möchte, ein leicht lesbares, gründliches und umfangreiches Lehrmaterial.
Gemäß dieser Zielrichtung haben die Autoren die PASCAL-Sprachbeschreibung mit einigen Ausführungen zum Wesen eines Rechners und dessen Programmierung sowie mit einem Hardware-Überblick umrahmt.
Die PASCAL betreffenden Abschnitte sind klar gegliedert, prägnant formuliert und geradezu verständniserzwingend dargestellt. Dabei bewährt sich wieder einmal die Methodik der Einheit von motivierender Globalerklärung, typischen Beispielen und Detailanalyse. Und auf diese Weise ist es auch nahezu mühelos gelungen, die allgemein etwas schwerer erlernbaren Teile wie die Behandlung von Datensatzstrukturen (Records), Files und dynamischen Variablen transparent zu machen. Allerdings werden vom etwas fortgeschrittenen Leser gerade an diesen Stellen umfassende syntaktische Schemata zur Unterstützung der Erklärungen vermißt. Sie werden jedoch in einem Anhang nachgereicht.
Das hier besprochene Buch legt den gesamten PASCAL-Sprachumfang gemäß User Manual and Report von Jensen/Wirth dar und berücksichtigt zudem die internationalen Standardisierungsbemühungen auf diesem Gebiet, soweit diese zum Zeitpunkt des Redaktionsschlusses zu konkreten Ergebnissen geführt hatten. Ein größerer Abschnitt am Ende befaßt sich mit den Spezifika von TURBO-PASCAL. Dadurch wird auf eine Implementation von ständig wachsender Verbreitung verwiesen. Der Anwender dieser Version kann jedoch auf umfangreichere Spezialliteratur nicht verzichten.

Prof. Dr. Jürgen Zaremba

UNIX als Basis für Softwareentwicklung

von T. Baggenstos, R. Marty, B. Mergler, P. Schnorf,
Springer Verlag Berlin – Heidelberg – New York – Tokyo 1986, 199 S.

Das zur Rezension vorgelegte Buch zu verschiedenen Aspekten der Arbeit mit dem Betriebssystem UNIX versteht sich als Arbeitshilfe, keinesfalls als Lehrbuch.
Wer bereits weiß, wie man programmiert und wie ein Betriebssystem arbeitet, braucht dafür keine Einführung mehr. Aber er braucht gleichsam eine Brücke zwischen seinen bisherigen Erfahrungen und den doch sehr gedrängten, knappen Spezialinformationen, die er in den UNIX-Systemunterlagen findet. Dieser Funktion wird das Buch vor allem durch die hohe Sachkenntnis der Autoren gerecht, die immer den Blick für die wesentlichen, an kaum einer anderen Stelle zu findenden Informationen offenhält.

Behandelt werden im einzelnen:
- UNIX-Systemübersicht mit den grundlegenden Datei-Prozeß- und Ein-/Ausgabekonzepten dieses Systems
- Beschreibung der UNIX-Benutzeroberfläche ‚shell' als Kommandointerpreter bzw. als Programmiersprache
- Eine sehr kurz gehaltene Übersicht über die Programmiersprache C
- Darstellung der wesentlichen Eigenschaften einiger Hilfsmittel zur Softwareentwicklung unter UNIX
- Beschreibung der Werkzeuge des UNIX-Systemprogrammierers.

Besonders gelungen sind dabei die Kapitel die sich mit der Benutzeradministration, mit der Dateisicherung/-wiederherstellung und mit der systemnahen Programmierung unter UNIX befassen. Der Leser findet hier eine Fülle nicht-trivialer, praxisnaher Programme und Beispiele, die besonders den sogenannten „UNIX-Superuser" interessieren.
Ins Auge fallen außerdem sofort die gut durchdachten, besonders anschaulichen Bilder, die dem Leser verschiedene Sachverhalte sehr übersichtlich präsentieren.
Die Schwäche des Buches ist ohne Frage die Tatsache, daß auf einem Umfang von etwa 200 Seiten jedes der genannten umfangreichen UNIX-Teilgebiete doch immer nur mehr oder weniger angerissen werden kann.

Dr. Ludwig Claßen

NEUERSCHEINUNGEN

Lokale Computernetze

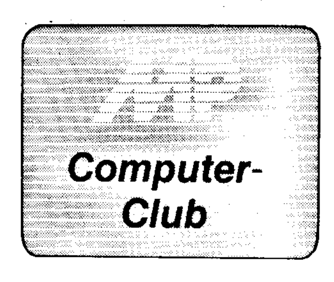
von B. Lindemann
Herausgeber: Kombinat Robotron
Etwa 224 Seiten, Broschur, DDR etwa 16,80 M, Ausland etwa 18,50 DM
Bestellangaben: ISBN 3-349-00241-2
676 080 6/Lindemann, Computern.
LSV 0395, 130/87, FM 87

Ein solides Grundwissen und ein Überblick über das gesamte Gebiet der lokalen Netze wird hier vom Autor vermittelt. Die Broschüre soll künftigen Nutzern lokaler Netze bei der Auswahl des geeigneten Netztyps und bei ähnlichen Fragen die Entscheidungen erleichtern. Ausführlich werden die Steuerverfahren für lokale Netze behandelt. Dabei werden die Leistungsfähigkeit sowie die Vor- und Nachteile der einzelnen Verfahren verdeutlicht. Beschrieben ist ferner eine Auswahl international bekannter Netze, unter besonderer Berücksichtigung von Entwicklungen in der DDR.

Betriebssystem SCP für Personalcomputer

von K. Schmidt
Herausgeber: Kombinat Robotron
Etwa 240 Seiten, Broschur, DDR etwa 15,00 M, Ausland etwa 15,00 DM
Bestellangaben: ISBN 3-349-00255-2
676 086 5/Betriebssystem SCP/PC
LSV 0393, 128/87

Die Schrift führt in das grundlegende Betriebssystem für den Personalcomputer PC 1715 ein. Sie vermittelt Kenntnisse für die umfassende und wirkungsvolle Nutzung dieser Gerätetechnik und bildet die Grundlage für das Verständnis und die unmittelbare Nutzung der SCP-kompatiblen leistungsfähigen Softwarekomponenten. Die Funktionsweise des Betriebssystems wird vor allem aus der Sicht des Systemprogrammierers (Aufbau und Funktionsweise) dargestellt. Die Besonderheiten des Betriebssystems SCP 1700 für den Arbeitsplatzcomputer A 7100 (16-Bit-Technik) werden kurz aufgeführt.



Unerklärliche Reaktionen der unter CP/M, CPA, SCP (X) ... lauffähigen Assembler M80 bzw. ASM

Nach Umsetzung eines ursprünglich für den U880-Cross-Assembler des KRS 4201 geschriebenen Programms auf den PC 1715 kam es bei der Assemblierung bei Befehlen der Art

```
JP    Z,MARKE
JR    Z,MARKE
CALL  Z,UP
RET   Z
JP    NC,MARKE
JR    NC,MARKE
CALL  NC,UP
RET   NC
```

zur Meldung des fatalen Fehlers „A". Tatsächlich war auch der erzeugte Operationscode falsch (z. B. 20H anstelle von 28H bei JR Z, ...).
Als Ursache wurde die Existenz von zwei Marken Z: und NC: im Programm ermittelt. Die entsprechenden Programmzeilen waren jedoch durch den Assembler nicht beanstandet worden.
Offensichtlich dürfen also die Zeichenketten, die Sprungbedingungen angeben (Z, NZ, C, NC, M, P, PE, PO) nicht als Marken verwendet werden, wenn mit den o. g. Assemblern gearbeitet werden soll.

Dr. Thomas Heinke

BASIC-Interpreter für Z1013

Seit einiger Zeit wird vom Robotron-Industrieladen Erfurt der an den MRB Z1013 angepaßte HC-BASIC-Interpreter auf Kassette vertrieben.
Zur Nutzung erreichten uns als Computer-Club zahlreiche Anfragen, die die Arbeit mit dem HC-BASIC bei einer RAM-Konfiguration von 64 KB betreffen.
Dazu von uns ein allgemeiner Hinweis:

- Nach dem Start des Interpreters mit J 300 ist auf die Abfrage der Speichergröße bei 64 KB-RAM als maximal nutzbare Adresse (dezimal) 60415 einzugeben und mit ENTER zu quittieren. Wird diese Einschränkung nicht vorgenommen, so belegt der BASIC-Interpreter den Bildwiederholspeicher und blockiert das System.

Abschließend sei noch die Anfrage an den VEB Robotron-Vertrieb Erfurt gestellt, warum jeder Interessent die Programmkassette in Erfurt abholen muß und kein Nachnahmeversand möglich ist.

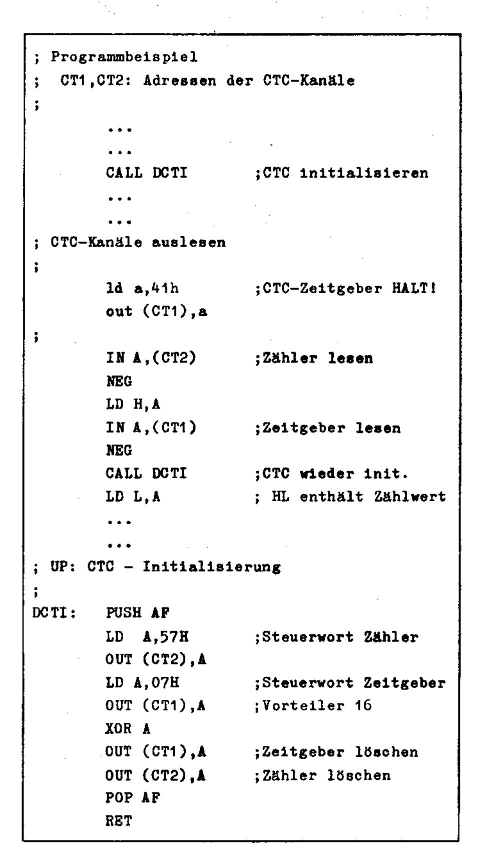
Hans-Jochen Bachmann
Computer-Club des Robotron-Anlagenbaus

Zu der Anfrage zum Vertrieb der Programmkassetten schrieb uns Kollege Daume, Direktor für Vertrieb des VEB Robotron-Vertrieb Erfurt:
An den Industrieladen des VEB Robotron-Vertrieb Erfurt ergeben sich nicht nur bezüglich der Programmkassetten, sondern auch bei anderen Waren Forderungen der Kunden nach einem Nachnahmeversand. Da die Verkaufseinrichtung – wie jedes andere Einzelhandelsgeschäft – nicht auf einen Versandhausbetrieb eingerichtet ist, kann der Vertriebsweg „Nachnahmeversand" nicht realisiert werden.
Als Entgegenkommen wird jedoch den Kunden weitgehend auch bei telefonischer Nachfrage eine Reservierung der im Angebot befindlichen Waren gewährleistet.

Zeitmessung mit kaskadierten CTC-Kanälen

Mit der IS U857, programmiert als Zeitgeber, können bei einer Systemtaktfrequenz von 2,5 MHz Ereignisse mit einer Genauigkeit von 6,4 µs ausgezählt werden. Sind die zu bestimmenden Zeitintervalle länger als 1,64 ms, müssen mehrere Kanäle der CTC mit Hilfe der Kanalausgänge ZC/TO kaskadiert werden. Hierbei arbeiten die nachgeschalteten Kanäle im Zählermode. Mit zwei Kanälen, Zeitgeber und Zähler, lassen sich bereits Zeitintervalle von maximal 420 ms messen. Problematisch erweist sich jedoch das Auslesen der kaskadierten Kanäle, da hierfür mehrere IN-Befehle benötigt werden. Ganz gleich, ob man zuerst den Zeitgeber oder den Zähler ausliest, in jedem Fall kann ein Übertrag während des Auslesens das Zählergebnis verfälschen. Die eindeutige Behandlung eines möglichen Übertrages kann softwaretechnisch nur durch mehrmaliges Auslesen und Vergleichen der entsprechenden CTC-Kanäle realisiert werden. Viel einfacher wird das Auslesen, wenn die Zählerkaskade steht. Dazu muß vom Programm der Zeitgeber angehalten werden, welches durch Setzen des Bits D1 (D1 = 1) im Kanalsteuerregister erreicht wird /1/. Leider wird in jedem Fall mit D1 = 1 der Kanal auch rückgesetzt, so daß ein nachfolgendes Auslesen des Zeitgeberzählerstandes nicht mehr möglich ist. Versuche mit verschiedenen Kanalsteuerwörtern ergaben, daß der gewünschte Effekt, Zeitgeber stoppen ohne rücksetzen, eintritt, wenn der Zeitgeber mit dem Steuerwort 41H in den Zählermode umprogrammiert wird. Ferner wurde festgestellt, daß der so umprogrammierte Kanal Impulse, die am Eingang CLK/TRG möglicherweise anliegen, nicht zählt. Wieder aktivieren läßt sich der CTC-Kanal mit einem Kanalsteuerwort, in dem D1 gesetzt ist. Das folgende Programmbeispiel benutzt zwei kaskadierte CTC-Kanäle und wurde an mehreren Schaltkreisen erprobt.

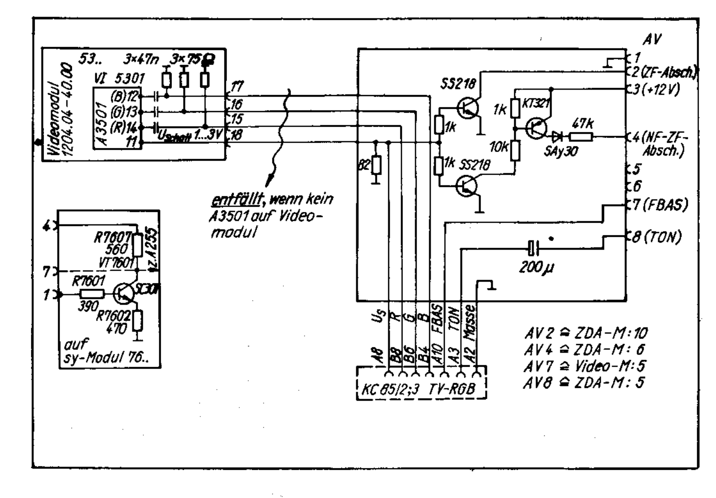
Christian Schiewe



```
; Programmbeispiel
;  CT1,CT2: Adressen der CTC-Kanäle
;
        ...
        ...
        CALL DCTI           ;CTC initialisieren
        ...
        ...
; CTC-Kanäle auslesen
;
        ld a,41h            ;CTC-Zeitgeber HALT!
        out (CT1),a
;
        IN A,(CT2)          ;Zähler lesen
        NEG
        LD H,A
        IN A,(CT1)          ;Zeitgeber lesen
        NEG
        CALL DCTI           ;CTC wieder init.
        LD L,A              ; HL enthält Zählwert
        ...
        ...
; UP: CTC - Initialisierung
;
DCTI:   PUSH AF
        LD  A,57H           ;Steuerwort Zähler
        OUT (CT2),A
        LD A,07H            ;Steuerwort Zeitgeber
        OUT (CT1),A         ;Vorteiler 16
        XOR A
        OUT (CT1),A         ;Zeitgeber löschen
        OUT (CT2),A         ;Zähler löschen
        POP AF
        RET
```

Literatur

/1/ Kieser, H.; Meder, M.: Mikroprozessortechnik, Berlin, VEB Verlag Technik, 1985, 3. Auflage



KC 85/2(/3)-Tip

Bild- und Tonanschluß mit RGB-Qualität für Farbfernsehgeräte der Serie 4000

Die Schaltung bietet bei Rechnereinschaltung die automatische Umschaltung von Fernsehbildwiedergabe auf Wiedergabe des vom Kleincomputer kommenden Bildes und Tones. Das Computerbild wird je nach Konfiguration des eingesetzten Fernsehgerätes in RGB- oder nur in FBAS-Qualität erzeugt. Benötigt wird ein Farbfernsehgerät der Serie 4000 (Colormat oder Colortron). Robotronfarbfernsehempfänger sind mit der auf diese Geräte zu modifizierenden Schaltung ebenfalls anschließbar.
Für RGB-Bildwiedergabe ist der VI 5301 IS A3501 auf dem Videomodul 1204.04-40.00/1, 2/ notwendig. Es müssen die 3 × 45-nF-Kondensatoren und die 3 × 75-Ohm-Widerstände zwischen Pin 12...14 der VI 5301 (Helligkeit des RGB-Bildes ist durch Widerstandswert beeinflußbar) und den Videomodulanschlüssen 15...17 gemäß dem Bild nachgerüstet werden. Zudem ist die Brücke zwischen Pin 11 der VI 5301 und Modulanschluß 18 zu legen. (Sämtliche Leiterzüge und Lötaugen sind auf dem Modul vorhanden.)
Das SY-Modul ist auf das Vorhandensein des VT 7601 ~~SC 307~~ zu überprüfen. Bei Nichtvorhandensein ist der Transistor mit seine Anschlußwiderständen gemäß dem Bild nachzurüsten. In diesem Fall entfällt der in ab 1986 gefertigten Fernsehempfängern statt seiner eingesetzte 120-Ohm-Widerstand von SY-Modul, Anschluß 7 in Richtung VI 7601 A255 /3/.
Die RGB- oder FBAS-Bild- und Tonansteuerung ist als Modul gefertigt und auf den auf der Gerätegrundplatte vorhandenen AV-Platz gesteckt (wenn Kontaktstifte nicht vorhanden sind, dann mit 1,5-mm-Cu-Drahtstücken nachrüsten), die RGB- und

SC 236

(Fortsetzung auf Seite 62)

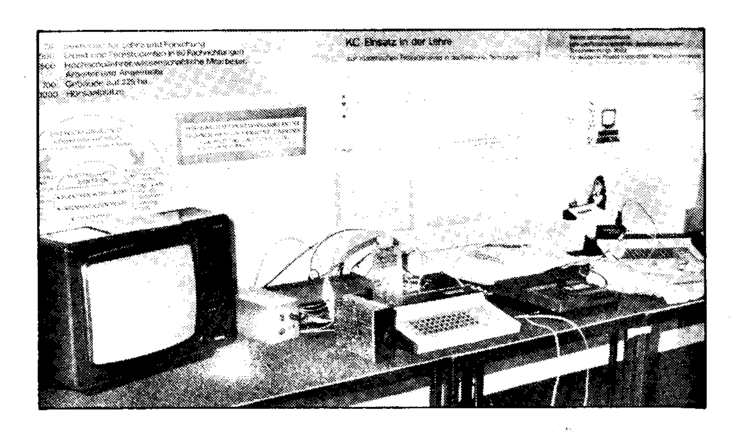


Bild 1 Das „Sieger"-Exponat – Floppy-Anschluß an KC 87

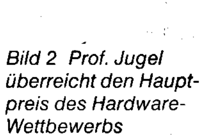




Bild 2 Prof. Jugel überreicht den Hauptpreis des Hardware-Wettbewerbs

Anläßlich des 750jährigen Berlinjubiläums fanden am 1. und 2. August vergangenen Jahres die „Dresdener Tage" statt, auf denen der Elbe-Bezirk sich und seine Leistungen vorstellte. Im Rahmen dieser Präsentation riefen der Kulturpalast Dresden, VEB Robotron-Meßelektronik Dresden und die TU Dresden, Informatikzentrum und Sektion Informationstechnik, zur 1. Programmierolympiade und zum Kleincomputer-Hardware-Wettbewerb auf.

Kleincomputer-Hardware-Wettbewerb

Der Leistungsvergleich stand unter dem Motto *Peripherie-Geräte für den KC 85/1 und den KC 87 aus Dresden.* Erwartungsgemäß fand der Wettbewerbsaufruf ein sehr breites Echo, wenn auch auf Grund der Vergleichbarkeit der Exponate, die bewußt beschränkte Thematik, nicht die Arbeiten aller „Computer-Freaks" in die Bewertung einbezog. Nach sorgfältiger Begutachtung der eingereichten Leistungen innerhalb der Laboratorien der Technischen Universität Dresden wählte eine Jury aus Mitarbeitern der Bildungseinrichtung und des Kombinats Robotron die vier besten Exponate (Floppy-Anschluß mit 64kdRAM-Erweiterung (Bild 1), programmierbarer Zeichengenerator, Einchip-Mikrorechner-Emulator und EPROM-Floppy) aus.
Innerhalb der Veranstaltung *simultan-wissenschaft live* am 1. und 2. August hatten die Berliner und ihre Gäste in der Kongreßhalle die Möglichkeit, die Besten des Wettbewerbs mit ihren Exponaten in Augenschein zu nehmen.
Zum Abschluß der Veranstaltung erhielt ein Studentenkollektiv aus den Händen des Betriebsdirektors des VEB Robotron Meßelektronik „Otto Schön", Prof. Dr. Jugel, den Hauptpreis, ein Mini-Floppy-Laufwerk (Bild 2). Der Wettbewerb sowie die Leistungsschau in Berlin bewiesen erneut das sehr breite Interesse in Sachen Computertechnik.

Eckhardt Herzmann

1. Programmierolympiade

Der beste Programmierer aus Studium, Lehre und Praxis, aus den Computerclubs und dem Kreis der Hobbyanwender war gesucht. Am 1. und 2. August 1987 sollten die Besten ihre Leistungen an Kleincomputern KC 87 des VEB Kombinat Robotron demonstrieren. Folgende Aufgabe war zu lösen:
Für eine Datenbasis waren eine Organisationsform für den Hauptspeicher und Algorithmen zu entwickeln und in

(Fortsetzung auf Seite 63)

(Fortsetzung von Seite 61)

US-Verbindung wird weiter auf das Videomodul gezogen. Das FBAS-Signal wird direkt vom TV-RGB-Port des KC über das Modul an AV7 geführt. Im Falle der Computer-RGB-Bildwiedergabe über den IS A3501 wird nur der an AV7 gelangende Synchronanteil benötigt. Bei Vorhandensein des Videomoduls 1204.00-40.00 (ohne IS A3501) entfällt die Signalweiterführung zum Videomodul. Das vom Computer an AV7 gelangende FBAS-Signal dient hierbei zusätzlich der Bildwiedergabe. Die Wiedergabe des Computerbildes in FBAS-Qualität steht der in RGB-Qualität nicht viel nach.
Die vom KC 85/2(/3) kommende Schaltspannung darf an Pin 11 des IS A3501 1 ... 3 V betragen. Sie steuert die Umschaltung des Fernsehgerätes auf Computerbild- und Tonwiedergabe bei gleichzeitiger Wegschaltung der zugehörigen TV-Signale. Bei anliegender Schaltspannung US wird AV2 auf Masse gelegt (Abschaltung des VI 7201 A241 auf ZDA-Modul); AV4 führt dann etwa 11 V (Abschaltung des Ton-ZF-Kanals VI 7202 A223 Pin 2). Die Einspeisung des Computertons erfolgt über AV8. Durch Zusätze zur Schaltung (Heruntersetzen der TTL-RGB-Signale auf 1 V gemäß den RGB-Eingangserfordernissen am Videomodul Pin 15; 16; 17, Verwendung des Signals BAS bzw. LUM zur Synchronisation, Erzeugen der Umschaltbedingung evtl. abgeleitet vom SYNC-Signal) sind auch Commodore-, Schneider- und andere Rechner mit RGB- bzw. nur FBAS-Peripherie über obige Schaltung an die genannten Farbfernsehgeräte anschließbar /5, 6/.

Egmont Blochwitz

Literatur

/1/ Schornack, W.; Wachsmuth, R.: Die RGB-Ansteuerung der Farbbildröhre mit der Videokombination A3501D. Radio, Ferns., Elektron., Berlin 35 (1986) 1, S. 31–34
/2/ Schornack, W.: Farbfernsehempfänger Colormat und Colortron der Serie 4000. Radio, Ferns., Elektron., Berlin, 34 (1985) 7, S. 439–444
/3/ Schornack, W.; Sonka, U.: Videorecorderanschluß für Farbfernsehempfänger. Radio, Ferns., Elektron., Berlin 35 (1986) 7, S. 439–442
/4/ System-Handbuch KC 85/3
/5/ Handbuch Commodore 128
/6/ Kundendienst Handbuch Schneider CPC 464

Erweiterter Service für Festkommazahlen für Kleincomputer

Wird der Computer für Berechnungen eingesetzt und werden die Ergebnisse ausgegeben, ist oft ein ungünstiges Druckbild zu bemängeln. Die Darstellung Komma unter Komma genügt z. B. bei Geldbeträgen nicht. Mit dem hier dargestellten Unterprogramm werden beliebige Zahlen als Festkommazahlen auf dem Drucker oder dem Bildschirm ausgegeben. Die Anzahl der gerundeten Nachkommastellen wird im Unterprogramm mit einer natürlichen Zahl (SZ), die größer als 0 sein sollte (für 0 kann einfacher die Integer-Funktion genutzt werden), festgelegt. Die zu formatierende Zahl wird mit dem Variablennamen Y in das Unterprogramm übergeben. Die formatierte Zahl steht unter Y$ zur Verfügung. Mit der numerischen Variablen H kann die formatierte Zahl in der Spalte Komma unter Komma ausgerichtet werden. Im Unterprogramm wird noch die Hilfsstring-Variable H$ zur Zwischenspeicherung benutzt.
Das Unterprogramm kann in weniger Zeilen, als hier angegeben, untergebracht werden und sollte, bei sehr häufigem Aufruf, noch vor dem Hauptprogramm untergebracht werden.

Hans-Jürgen Gatsche

```
10 REM *** TESTRAHMEN ***
20 WINDOW 0,31,0,39 : COLOR 7,1 : CLS : WINDOW
30 FOR X = -10 TO 10 STEP .5
40 Y = RND(1) * X : REM  ZU FORMATIERENDE ZAHL
50 GOSUB 1000 : PRINT TAB(20-H); Y$
60 NEXT : END
999 REM *** UP  PRINT USING - ERSATZ  XXXX.XX ***
1000 SZ = 2 : REM  NACHKOMMASTELLENZAHL >1
1010 Y = INT(Y * 10^SZ +.5)/10^SZ : Y$ = STR$(Y)
1020 IF INSTR(".",Y$) = 0 THEN Y$ = Y$ + ".000000" : ELSE Y$ = Y$
+ "00000"
1030 H = INSTR(".",Y$) : Y$ = LEFT$(Y$,H+SZ) : H$  = RIGHT$(Y$,
LEN(Y$)-H)
1040 IF INT(Y) = 0 AND Y > 0 THEN Y$ = "0." + H$
1050 IF INT(Y) =< 0 AND Y > -1 AND Y < 0 THEN Y$ = "-0." + H$
1060 H = INSTR(".",Y$)
1070 RETURN
```



Bild 3 *55 Einsender erhielten die Möglichkeit der Teilnahme am Vor- und Endausscheid in Berlin*



Bild 4 *Ralf Krebs mit dem 1. Preis der Programmierolympiade – einem KC 87*

Fotos (4): Paszkowsky

BASIC zu implementieren. Die folgenden Operationen sollten dabei so realisiert werden, daß die erforderlichen Suchvorgänge auch bei großen Datenbeständen sehr schnell ausgeführt werden:

- Einlesen der Datenbasis, die als Datei auf Magnetbandkassette unsortiert vorliegt und Aufbau der Datenstrukturen im Hauptspeicher. Die auf Magnetband befindliche Datenbasis war zu dimensionieren mit DA$ (200,2).
- Einlesen eines Schlüsselwortes von der Tastatur.

Ist ein Satz mit diesem Schlüssel in der Datenbasis, wird er auf dem Bildschirm angezeigt.
Der Nutzer hat die Möglichkeit
– den Informationsteil zu ändern und in die Datenbasis zu speichern
– den Satz zu löschen.
Ist der Satz mit diesem Schlüssel nicht in der Datenbasis, kann der Nutzer
– die Operation beenden
– den Informationsteil eingeben und den Satz in die Datenbasis einfügen.

- Ausgabe der geänderten Datenbasis auf Magnetbandkassette.

Ein Satz hat folgenden Aufbau:
Schlüssel:
Zeichenkette aus max. 8 Buchstaben, evtl. mit Leerzeichen aufgefüllt
Informationsteil:
Zeichenkette aus max. 20 Zeichen
Maximale Anzahl der Sätze: 200

Die beschriebene Aufgabe war mit einem Robotron-Kleincomputer zu lösen. Die Lösung hatte folgende Teile zu umfassen:

- BASIC-Programm auf Magnetbandkassette
- Dokumentation (max. 5 Seiten A4)

– Nutzerbeschreibung
– Kurzdarstellung der Lösungsidee
– Darstellung der Programmstruktur
– Beschreibung der Teststrategie

Bewertet wurden in der Reihenfolge

- Korrektheit der Lösung
- Klarheit und Verständlichkeit von Programmen und Dokumentationen
- Originalität der Lösungsidee
- Effizienz der Lösung (Rechenzeit und Speicherbedarf)

Insgesamt beteiligten sich etwa 520 Jugendliche im Alter von 14 bis 26 Jahren an diesem Wettbewerb. Nach einer Vorauswahl erhielten 55 Einsender die Möglichkeit der Teilnahme am Vor- und Endausscheid in Berlin. In Berlin erhielten alle Teilnehmer eine geänderte Aufgabenstellung, und nach einer Vorbereitungszeit waren

– das mitgebrachte Programm in den Rechner einzugeben
– die Änderungen am eigenen Programm auszuführen, die der gestellten Aufgabe gerecht werden und
– die Korrektheit des geänderten Programms anhand von Testdaten der Wettbewerbskommission zu überprüfen.

Diejenigen Teilnehmer, die diese Aufgabe am schnellsten lösten, kamen in den Endausscheid. Von der Kommission wurde unter Ausschluß des Rechtsweges die Rangfolge der Lösungen einschließlich ihrer Dokumentation festgelegt.
Die Auswahl des Siegers erfolgte im Endkampf nicht mehr nach dem Gesichtspunkt des effektiv gestalteten Programmes, sondern es war die Beherrschung der Programmiersprache, der Technik und der eigenen Lösung zu beweisen.
Das Programm des Siegers, Ralf Krebs, Neubrandenburg, stellt damit nicht zwingend die optimale Lösung dar, es hatte aber die besten Eigenschaften hinsichtlich der Forderung nach einer Änderbarkeit.
In Bild 5 (siehe S. 64) ist das Programm in der Originalfassung wiedergegeben. Für die Nutzung sind die einzelnen Module durch ein geeignetes Menübild mit entsprechenden Auswahlmöglichkeiten zu verbinden.
Ein Überblick über die einzelnen Programmteile wird durch Bild 6 vermittelt.

Prof. Dr. Rolf Giesecke,
Dr. L. Schönfelder

Programm KREBS

1. Nutzerbeschreibung

1.1. Vorbemerkungen

Das Programm wurde an Kleincomputern KC 85/1 (BASIC- und 1 RAM-Modul) und KC87 (1 RAM-Modul) entwickelt. Es benötigt jedoch den zusätzlichen RAM-Modul nicht.

1.2. Programm einlegen und Programmstart

Das Programm wird mit:
CLOAD "KREBS"
in den Computer eingelesen. Anschließend kann es sofort ohne weitere Modifikationen durch Betätigung der Taste:
RUN gestartet werden.

1.3. Weiterer Ablauf

Der weitere Ablauf wird durch die Ausgabe von entsprechenden Mitteilungen gesteuert. Als erstes wird z. B. der Name der zu bearbeitenden Datenbasis abgefordert. Bei der Eingabe von variabel langen Zeichenketten (Name der Datenbasis, Schlüsselwörter Informationsteile) sind keine begrenzenden Hochkomma erforderlich. Die Eingabe wird stets durch Betätigung der Taste ENTER abgeschlossen.
Eine Ausnahme bildet der Fall, wenn bei Eingabe der Schlüsselworte oder der Informationsteile die maximal erlaubte Länge erreicht wird. In diesem Fall wird die Eingabe automatisch beendet.

2. Kurzdarstellung der Lösung

Um einen schnellen Zugriff auf die Daten zu gewährleisten, wird das Verfahren binäres Suchen eingesetzt. Es findet einen beliebigen Satz unter 200 Sätzen in maximal 8 Suchschritten. Das binäre Suchen setzt einen sortierten Datenbestand voraus. Somit ergibt sich folgender Grobablauf:

① Einlesen der Datenbasis
② Sortieren der Datenbasis
③ Bearbeiten im Sinne der Aufgabenstellung
④ Ausgabe der geänderten Datenbasis.

Um das Warten beim ehemaligen Sortieren so gering wie möglich zu halten, wurde ein QUICK-Sort-Verfahren eingesetzt. Es bezieht seine Leistungsfähigkeit im wesentlichen aus drei Punkten:

① Indizierung der Datenelemente
② Ringtausch
③ Vorsortierung.

Aufgrund der Besonderheiten von BASIC-Interpretern wurden die Sortierung und das binäre Suchen am Anfang des Programms in komprimierter Form eingegeben.
Des weiteren wurden aus der Aufgabenstellung mehrere Einschränkungen abgeleitet:

① Kein Assemblerunterprogramm (Programmiersprache: BASIC)
② DA$(0,m) und DA$(n,0) bleiben leer (200 Datenelemente mit je 2 Bestandteilen und DIM DA$ (200,2) [m = 0...2, n = 0...200]
③ Es wird aufgrund fehlender Vorgaben ein ab DA$(1,m) lückenlos abgespeicherter Datenbestand vorausgesetzt
④ Die geänderte Datenbasis wird aufgrund fehlender Vorgaben unsortiert wieder ausgegeben.

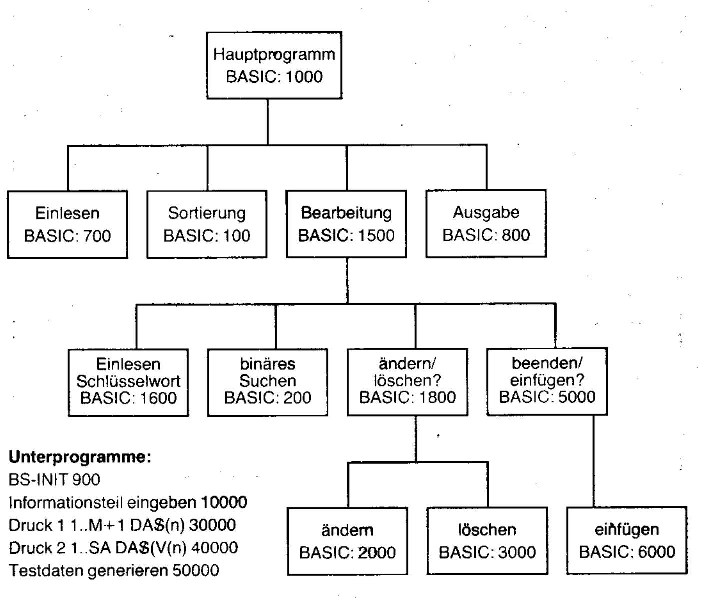


Bild 6 *Programmstruktur*

```
LIST

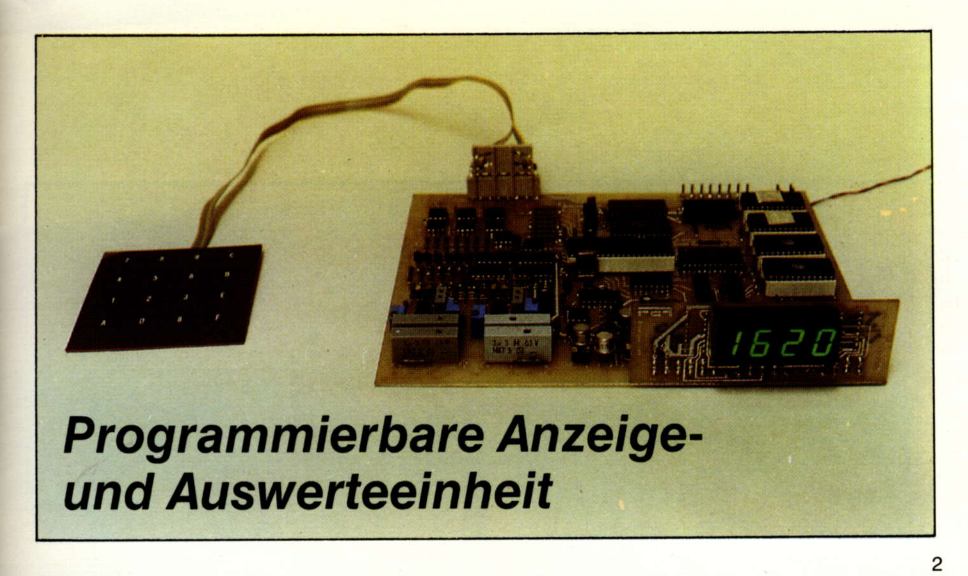
10 GOTO1000
100 FORI=2TON:IFDA$(I,2)<>""THENV(I)=I:NEXT:ELSES=I-1:SA=S
110 S=INT(S/2):IFS=0THENRETURN
120 FORI=S+1TOSA:W=V(I)
130 FORJ=ITOS+1STEP-S:IFDA$(V(J-S),2)<=DA$(W,2)THEN150
140 V(J)=V(J-S):NEXTJ
150 V(J)=W:NEXTI:GOTO110
200 OB=SA:UN=1 :NR=0
210 MI=INT((OB+UN)/2):IF UN>OB THENRETURN
220 IF ZW$<DA$(V(MI),1) THEN OB=MI-1 :GOTO210
230 IF ZW$>DA$(V(MI),1) THEN UN=MI+1 :GOTO210
240 NR=MI
250 RETURN
400 DA$(0,0)=STR$(V(1))
401 M=V(1)
410 FOR I=1 TO SA-1
420 DA$(M,0)=STR$(V(I+1))
425 M=V(I+1)
430 NEXT
435 DA$(M,0)="0"
440 !
450 M=   VAL(DA$(0,0) )
460 PRINT DA$(M,2),DA$(M,1):IF DA$(M,0)="0"THEN RETURN
475 M=VAL(DA$(M,0))
490 GOTO460
700 !EINLESEN
705 INPUT "1. EINGABE DATEINAME: ";F$:PRINT
710 PRINT "2. BAND AUF DATENBESTAND POSITIONIEREN" :PRINT
720 PRINT "3. WIEDERGABE STARTEN" :PRINT
730 PRINT "4. BEI ERTOENEN DES VORTONS BITTE DIE"
740 PRINT "    TASTE: ENTER BETAETIGEN.":PRINT
750 IF INKEY$<>CHR$(13)THEN750
760 PRINT "5. BITTE WARTEN":PRINT
765 !GOSUB50000
770 CLOAD*""+F$;DA$
780 RETURN
800 ! AUSGABE
810 PRINT"1. BAND POSITIONIEREN":PRINT
820 PRINT"2. AUFNAHMETASTEN BETAETIGEN":PRINT
830 PRINT"3. DANACH SOFORT DIE TASTE: ENTER"
835 PRINT"    BETAETIGEN":PRINT
836 IF INKEY$<>CHR$(13)THEN836
840 PRINT"4. BITTE WARTEN!":PRINT
850 CSAVE*""+F$;DA$
860 RETURN
900 !BS-INIT
910 WINDOW0,19,0,39:OUT136,4:PAPER1:INK3:CLS
920 PRINT" --- WIR BEHERRSCHEN DEN COMPUTER     "
930 PRINT"            TEILNEHMER: RALF KREBS":PRINT
940 RETURN
1000 !HAUPTPROGRAMM
1010 CLEAR6100
1020 J=0:N=200:DIM DA$(N,2),V(N)
1030 GOSUB900:GOSUB700:!EINLESEN
1040 V(1)=1:S=N:SA=N
1050 GOSUB900:PRINT"    BITTE WARTEN":PRINTAT(4,0);" ":GOSUB1
00:!SORT
1060 GOSUB400:!NEU
1070 GOSUB900:GOSUB800:!AUSGEBEN
1080 GOSUB900
1090 PRINT"RECORDER STOPPEN!":PRINT
1100 PRINT"              AUF WIEDERSEHEN"
1110 END
1500 !BEARBEITEN
1510 GOSUB1600:!EINLESEN SCHLUESSEWORT
1520 IF ZW$="" THEN RETURN
1530 GOSUB200:!BINAERES SUCHEN
1540 IF NR=0 THEN GOSUB5000:ELSE GOSUB1800
1550 GOTO1510
1600 ! EINLESEN SCHLUESSELWORT
1610 GOSUB900:!BS INIT
1611 PRINT"KLEINSTER SCHLUESSEL: ";DA$(V(1),1)
1612 PRINT"GROESSTER SCHLUESSEL: ";DA$(V(SA),1):PRINT
1613 PRINT"ANZAHL DER SAETZE:";SA:PRINT
1620 PRINT"EINGABE EINES SCHLUESSELWORTES:"
1630 PRINT"(ENTER OHNE WORT BEENDET DIE EINGABE!)"
1640 ZW$ = ""
1650 WINDOW 12,12.15,22:CLS
1660 FOR I=1 TO 8
1670 TA$ = INKEY$
1680 IF TA$="" THEN1670
1690 IF TA$=CHR$(13) THEN1740
1700 IF TA$<" " THEN1670
1710 ZW$ = ZW$+TA$
1720 PRINT TA$;
1730 NEXT
1740 RETURN
1800 ! AENDERN/LOESCHEN?
1810 GOSUB900:!BS INIT
1820 PRINT"SCHLUESSELWORT: ";ZW$.:PRINT
1830 PRINT"INFORMATIONSTEIL: ";DA$(V(NR),2):PRINT
1840 PRINT"    1 - AENDERN":PRINT
1850 PRINT"    2 - LOESCHEN"
1860 PRINTAT(7.0);" "
1870 TA$ = INKEY$
1880 IF TA$<"1" OR TA$>"2" THEN1870
1890 ON VAL(TA$) GOSUB 2000,3000
1900 RETURN
2000 ! AENDERN
2010 GOSUB10000:!INFORMATIONSTEIL EINGEBEN
2020 DA$(V(NR),2)=IT$
2030 RETURN
3000 ! LOESCHEN
3005 PRINT:PRINT"BITTE WARTEN":PRINTAT(11,0);" "
3010 SA=SA-1
3020 GR=V(NR)
3030 FOR I=GR TO SA
3040 DA$(I,1)=DA$(I+1,1)
3050 DA$(I,2)=DA$(I+1,2)
3060 NEXT
3070 DA$(SA+1,1)=""
3080 DA$(SA+1,2)=""
3090 FOR I=NR TO SA
3100 V(I)=V(I+1)
3110 NEXT
3120 FOR I=1 TO SA
3130 IF V(I)>GR THEN V(I)=V(I)-1
3140 NEXT
3150 RETURN
5000 ! BEENDEN/EINFUEGEN
5010 GOSUB900:!BS-INIT
5020 PRINT"SCHLUESSEL: ";ZW$;" NICHT GEFUNDEN":PRINT
5030 PRINT"    1 - OPERATION BEENDEN":PRINT
5040 PRINT"    2 - SATZ EINFUEGEN"
5050 PRINTAT(8,0);" "
5060 TA$ = INKEY$
5070 IF TA$<"1" OR TA$>"2" THEN5060
5080 ON VAL(TA$) GOSUB5090,6000
5090 RETURN
6000 !EINFUEGEN
6010 IF SA<N THEN6070
6020 PRINT:PRINT"TABELLE VOLL, EINFUEGEN UNMOEGLICH":PRINT
6030 PRINT"BITTE BETAETIGEN SIE EINE TASTE!"
6040 PRINTAT(7,0);""
6050 IF INKEY$=""THEN6050
6060 RETURN
6070 GOSUB10000:!INFORMATIONSTEIL EINGEBEN
6080 SA = SA+1
6090 FOR I=SA TO UN+1 STEP -1
6100 V(I)=V(I-1)
6110 V(I)=V(I-1)
6120 NEXT
6130 DA$(SA,1)=ZW$
6140 DA$(SA,2)=IT$
6150 V(UN)=SA
6160 RETURN
10000 !INFORMATIONSTEIL EINGEBEN
10010 GOSUB900:!BS INIT
10020 PRINT"SCHLUESSELWORT:":ZW$:PRINT
10030 PRINT"INFORMATIONSTEIL EINGEBEN:"
10040 PRINT"(MAXIMAL 20 ZEICHEN)"
10050 WINDOW8,8,10,27:CLS
10060 IT$=""
10070 FOR I=1 TO 20
10080 TA$=INKEY$
10090 IF TA$="" THEN10080
10100 IF TA$=CHR$(13) THEN10150
10110 IF TA$<" " THEN10080
10120 PRINTTA$;
10130 IT$=IT$+TA$
10140 NEXT
10150 RETURN
30000 !KONTROLLDRUCK1
30010 FOR I=1 TO M+1
30020 PRINTI;DA$(I,1),DA$(I,2)
30030 NEXT
30040 RETURN
40000 !KONTROLLDRUCK2
40010 FOR I=1 TO SA
40020 PRINTI;V(I);DA$(V(I),1);" ";DA$(V(I),2)
40030 NEXT
40040 RETURN
50000 ! TESTDATEN GENERIEREN
50010 M=10
50020 FOR I=1 TO M
50030 ZW$=STR$(I):PRINT I;
50040 DA$(I,1)=STRING$(4-LEN(ZW$),"0")+MID$(ZW$,2)
50050 DA$(I,2)="TEXT:"+MID$(ZW$,2)
50060 NEXT I:PRINT
50070 FOR I=1 TO 5*M:PRINT I;
50080 X=INT(M*RND(1)+1)
50090 Y=INT(M*RND(1)+1)
50100 ZW$=DA$(X,1)
50110 DA$(X,1)=DA$(Y,1)
50120 DA$(Y,1)=ZW$
50130 ZW$=DA$(Y,2)
50140 DA$(Y,2)=DA$(X,2)
50150 DA$(X,2)=ZW$
50160 NEXT I:PRINT
50170 !CSAVE*"DA";DA$
50180 !GOSUB30000:PAUSE
50190 RETURN
OK
```

Bild 5 Programmausdruck

3. Teststrategie

Die einzelnen Module wurden entsprechend der Programmstruktur von oben nach unten entwickelt und getestet. Fehlende Module wurden zunächst durch einfaches RETURN ersetzt. Zum Test wurden zunächst weiter die Module ab 30000 genutzt, die nicht zum eigentlichen Programm gehören. Da sie eine geringe Größe haben, wurden sie im Programm belassen. Das bringt bei späteren Änderungen Vorteile.

Durch Auswahl der Testdaten mit Hilfe des Moduls ab 50000, konnten die verschiedensten Testdaten gebildet werden. Die eingegebenen Schlüsselworte wurden so gewählt, daß möglichst alle Wege im Programm durchlaufen werden. Des weiteren wurde ständig überprüft, ob die Forderungen der Aufgabenstellung streng erfüllt werden und eine einfache und sichere Abarbeitung gegeben ist.



Programmierbare Anzeige- und Auswerteeinheit

2

Bernd Möller
VEB Forschung, Entwicklung und Rationalisierung des Schwermaschinen- und Anlagenbaus Magdeburg

Vorbemerkung

In zahlreichen technologischen Prozessen verlangt die Analyse der Prozeßsignale eine mikrorechnergestützte Auswertung. Vorteilhaft ist vielfach ein speziell zugeschnittenes und damit leicht handhabbares Gerät.

Die programmierbare Anzeige- und Auswerteeinheit (PAA) erfüllt insbesondere die Erfordernisse zum Anschluß an einen Hochtemperatursensor zur Überwachung von Stahlschmelzen. Ausgewertet werden die Thermospannung an einem Halbleiter unter dem Einfluß eines Temperaturgradienten, die etwa 500 mV betragen kann, und das Signal eines Fühlerkopftransmitters in Verbindung mit einem Widerstandsthermometer /1/, /2/. Aus diesen beiden Signalen und einem über eine Tastatur eingegebenen Vergleichswert ermittelt die PAA die Temperatur der Stahlschmelze kontinuierlich.

Das Gerät ist so konzipiert, daß es eine Vielzahl meßtechnischer Probleme löst, und es kann durch programmtechnische Änderungen an die unterschiedlichsten Meßaufgaben angepaßt werden /3/.

Verwendungszweck

Die PAA ist eine universelle Mikrorechnerlösung auf der Basis eines EMR zur Erfassung, Verarbeitung und Anzeige von Prozeßsignalen. Als analoge Signale können Spannungen von −99 bis +999 mV oder Ströme von 0 bis 20 mA bzw. 4 bis 20 mA erfaßt werden.

Zwei vom Rechnerkern potentialgetrennte ADU stellen dem EMR die Meßwerte in einer verarbeitungsgerechten Form zur Verfügung. Diese Meßwerte können vom EMR nach vorgegebenen Algorithmen verarbeitet werden. Zu den Aufgaben des EMR zählen u. a. die Bildung von Meßwertdifferenzen, -mittelwerten, -tendenzabschätzungen, Bestimmung von zeitlichen Änderungen. Die Anzeige dient zur visuellen Ausgabe dieser Größen. Zur zusätzlichen Eingabe kann eine Tastatur verwendet werden. Außerdem gestattet die PAA die Kopplung mit weiteren Geräten gleicher Priorität oder mit übergeordneten Rechnern zur zentralen Meßwerterfassung und -verarbeitung. Zusätzlich ist ein 8-Bit-paralleler Ausgang vorhanden, über den der Anwender frei verfügen kann, beispielsweise zum Anschluß eines DAU in Verbindung mit einem Schreiber. Bild 1 zeigt die Struktur der Verarbeitungselektronik.

Aufbau

Die wesentlichen Strukturen einer komfortablen Mikrorechnerkonfiguration sind bereits im EMR enthalten. Dazu zählen der interne RAM-Bereich, der Timer, das serielle E/A-Port und die parallelen Schnittstellen.

Durch Aufrüstung wird die Mikrorechnerkonfiguration zu einem kompakten Gerät (Bild 2). Über den EMR hinaus sind folgende Baugruppen vorhanden:

- externer Speicher
  - 8 KByte EPROM
  - 1 KByte RAM
- 2 Analog-Digital-Wandler
- 1 4×4-Matrix-Tastatur
- 1 4stellige Anzeige
- 1 8-Bit-paralleler Digitalausgang
- Logik-Baugruppe
- Stromversorgung

sowie durch schaltungstechnische Anpassung an Standard-Schnittstellen

- 1 serieller Digitaleingang (IFSS)
- 2 serielle Digitalausgänge (IFSS).

Bild 3 zeigt als Blockschaltbild das funktionelle Zusammenwirken der einzelnen Baugruppen des Gerätes.

Die Basis des Gerätes bildet ein EMR vom Typ U 883. Dieser arbeitet mit einer externen Taktfrequenz von 4,194 MHz. Port 0 und 1 stellen die Speicherbezüge her. Als Tastaturport ist Port 2 festgelegt, da dieses Port bitprogrammierbar ist. Port 3 stellt die Ein- und Ausgänge für die Timer-Kanäle (T_{in}, T_{out}) und für den SIO-Kanal (S_{in}, S_{out}) sowie die Signale zur Auswahl der seriellen Digitalausgänge (P 35) und zur Auswahl der Analogeingänge (P 34) zur Verfügung. Es besitzt somit einen universellen Charakter.

Das Gerät besitzt zwei Analogeingänge, die vom Rechnerkern potential getrennt sind. Eine Entkopplung der beiden Eingänge untereinander ist nicht vorhanden. Die Umwandlung der analogen Signale erfolgt mit einem ADU vom Typ C 520. Dieser legt auch die Kennwerte der Umsetzung fest. Der Zugriff auf die gewandelten Analogsignale erfolgt über eine Speicheroperation im externen Speicher des EMR. Der ADU wird somit im Memory-mapping betrieben und es können die Befehle LDC und LDE verwendet werden. Als Speicherplatz ist die Adresse 5000H festgelegt.

Die serielle Dateneingabe ist für die Kopplung mit weiteren Anzeige- und Auswerteeinheiten oder mit übergeordneten Rechnerkonfigurationen vorgesehen. Sie erfolgt über zweiadrige Stromschleifen, die vom Rechnerkern potentialgetrennt sind. Gleiches gilt für die serielle Datenausgabe, wobei diese auch zur Kopplung mit peripheren Geräten (Fernschreiber, Drucker) genutzt werden kann.

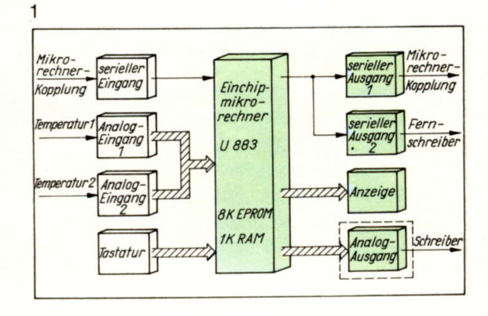
Zur Eingabe von Meß- und Korrekturwerten sowie zum Aufruf von Hilfs- und Testprogrammen befindet sich auf dem Mikrorechnerkern eine Schnittstelle zum Anschluß einer 16teiligen Folientastatur. Sie ist auf ein Minimum konzipiert, um die Bedienerhandlungen weitestgehend einzuschränken bzw. effektiv zu gestalten.

Die Anzeige der Prozeßdaten erfolgt über eine 4stellige 7-Segmentanzeige. Den Zugriff erreicht der EMR über zwei externe Speicheroperationen, in gleicher Art, wie sie im Zusammenwirken mit dem ADU ablaufen. Als Speicherplätze sind 4000 H und 4800 H festgelegt.

Gleiches gilt für das 8-Bit-parallele Digitalausgabeport, welches für den Anwender frei verfügbar ist. Dessen Speicherplatz ist mit 5800 H festgelegt.

Im Gerät werden drei Spannungen benötigt. Für den Rechnerkern und für die Analogeingänge 5 V, die aufgrund der Potentialtrennung getrennt bereitgestellt werden, sowie 12 V für die seriellen Digitalausgänge.

1



3

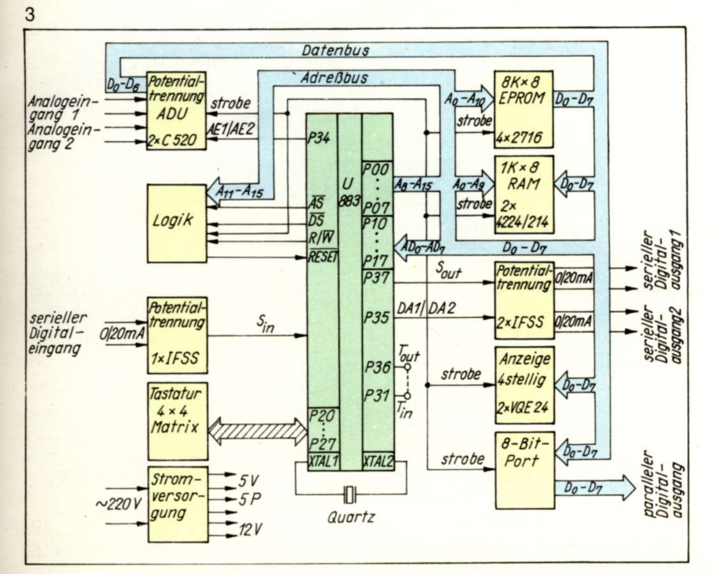


Bild 1 Struktur der Verarbeitungselektronik

Bild 2 Ansicht der Anzeige- und Verarbeitungselektronik

Bild 3 Übersicht der PAA

Literatur

/1/ Schulz, P.; Möller, B.: Kontinuierliche Temperaturüberwachung von Stahlschmelzen, msr, Berlin 30 (1987) 3 S. 98–102.

/2/ Möller, B., Schulz, P.: Praktische Untersuchungen zur kontinuierlichen Temperaturmessung in Stahlschmelzen, msr, Berlin 30 (1987) 7 S. 313–317

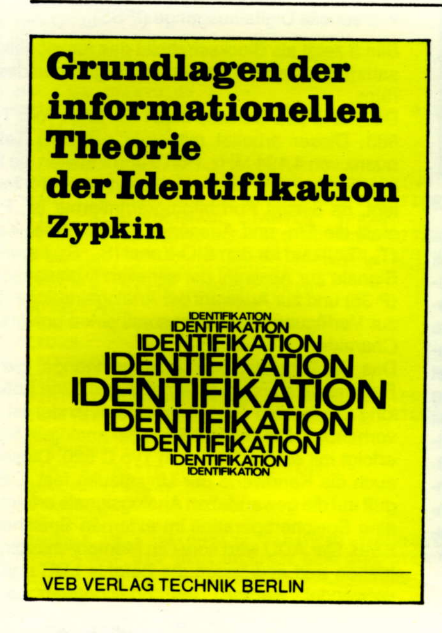
/3/ Möller, B.: PAA-Programmierbare Anzeige- und Auswerteeinheit; Dokumentation VEB FER Magdeburg 1987

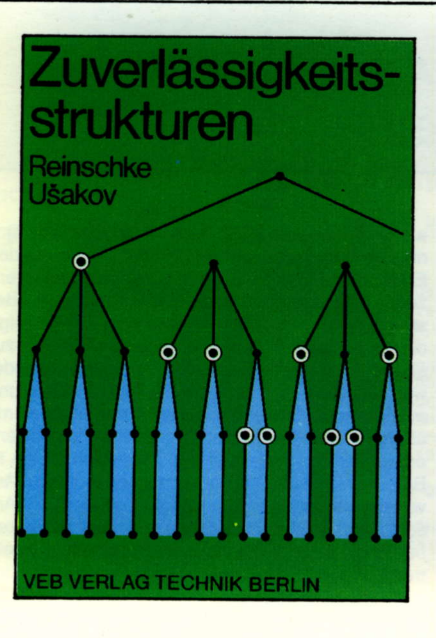
☒ KONTAKT ☏

VEB Forschung, Entwicklung und Rationalisierung des SAB, Abt. SEZ, Bleckenburgstr. 25, Magdeburg, 3011; Tel. 4 42 81

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 2
S. 33–64

Neuerscheinungen







Grundlagen der informationellen Theorie der Identifikation

Von Prof. Dr. d. techn. Wiss. Ja. S. Zypkin. Aus dem Russischen. 255 Seiten, 92 Bilder, 13 Tafeln, Kunstleder, DDR 36,– M, Ausland 56,– DM. Bestellangaben: 553 761 0/Zypkin, Identifikation

Der Autor zeigt mit seiner informationellen Theorie der Identifikation eine Möglichkeit, geeignete Verfahren auf der Grundlage vorhandener A-priori-Information auszuwählen. Damit wendet er sich sowohl an Theoretiker auf den Gebieten der Kybernetik, mathematischen Statistik und angewandten Systemanalyse, die sich mit der Identifikation dynamischer Systeme beschäftigen, als auch an Praktiker, die in Technik, Medizin, Biologie und anderen Bereichen diese Algorithmen anwenden können.

Zuverlässigkeitsstrukturen

Modellbildung · Modellauswertung
Von Dr.-Ing. habil. Dr. rer. nat. Kurt Reinschke und Prof. Dr. sc. techn. I. A. Ušakov. 240 Seiten, 95 Bilder, 14 Tafeln, Leinen, DDR 33,– M, Ausland 38,– DM. Bestellangaben: 553 728 2/ Reinschke, Zuverlässig

Hauptanliegen dieses Buches ist die Modellbildung und Untersuchung der Zuverlässigkeitsstruktur von Systemen. Als das entscheidende Darstellungsmittel werden Graphen benutzt. Die Methoden sind sowohl auf technische als auch auf nichttechnische Systeme anwendbar.
Die Darstellung ist weitgehend elementar, es werden keine Vorkenntnisse über Zuverlässigkeitstheorie vorausgesetzt.

Auslieferung durch den Fachbuchhandel

Prozeßanalyse mit unscharfen Verfahren

Von Dr. sc. techn. Steffen F. Bocklisch. 196 Seiten, 131 Bilder, 7 Programme, Leinen DDR 27,– M, Ausland 37,– DM. Bestellangaben: 553 746 9/Bocklisch, Prozeßanalyse

Vermittelt wird eine Methode, mit der die Modellbildung für große Systeme möglich wird, wie sie in Technik, Medizin, Ökonomie oder Umwelt zu finden sind. Die Besonderheit liegt darin, daß Unsicherheiten gemessener Daten erfaßbar sind und sprachlich formulierte Aussagen des Fachmanns verarbeitet werden können. Die unscharfe Prozeßanalyse ist in der Lage, Grob- und Feinmodelle zu entwickeln sowie Steuerentscheidungen abzuleiten. Mit der Einführung in diese Theorie und mit praktischen Beispielen (bis hin zu Rechnerprogrammen) werden potentielle Anwender und Interessenten aus vielen technischen und nichttechnischen Bereichen angesprochen.



VEB VERLAG TECHNIK BERLIN

ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK

MP

Heft 3 · 1988

Mikroprozessortechnik

VEB Verlag Technik Berlin

ISSN 0233-2892

Videosteuerung mit dem GDC U82720D



- Moderne Mikrorechnersysteme
- Bauanleitung für RAM-Disk



Symposium über Videoton-Computertechnik

Vom 16. bis 20. November 1987 veranstaltete die ungarische Videoton AG in Erfurt im Hotel Erfurter Hof ein rechentechnisches Symposium. Experten der Videoton AG informierten auf dieser Veranstaltung in Fachvorträgen über neueste Entwicklungen des Unternehmens auf den Gebieten Hardware und Software. Im Tagungsprogramm war Zeit für spezielle Konsultationsgespräche vorgesehen, in denen Nutzer von Videoton-Leistungen ihre Fragen und Probleme klären konnten.
Außerdem beriet während des Symposiums die R11-Nutzergemeinschaft über ihre Arbeit.
Besonderes Interesse aller Teilnehmer galt dem ausgestellten VT32-System, mit Motorola-kompatiblem 16/32-Bit-μP, das durch im System integriertem Arithmetikprozessor bei der Demonstration erstaunliche Ergebnisse für CAD-Aufgaben zeigte. Der VT 32 wird mit 1 bis 4 MByte Hauptspeicherkapazität ausgeliefert. Er hat eine Verarbeitungsgeschwindigkeit von 0,5 Mips. Die Systeme lassen sich auf Ethernet-Basis vernetzen.
An Anwendersoftware für dieses System wurde u. a. ein Paket zur Konstruktion von Rohrleitungen vorgestellt. Während Prototypen für das Nachfolgemodell VT320 mit 1 Mips bereits existieren, steht der VT3200 mit 32-Bit-μP, 2,0 Mips, virtuellem und Cache-Speicher noch auf dem Reißbrett.
Den VT32 werden wir in einer späteren Ausgabe noch ausführlicher beschreiben.
Für Nutzer des R11 wurden Möglichkeiten aufgezeigt, wie PCs von Robotron und anderen Herstellern an den Rechner angeschlossen werden können.
Das nächste Symposium soll vom 3. bis 6. Mai 1988 in Klingemühle, Bezirk Frankfurt/Oder, stattfinden. *MP*

KONTAKT

Videoton AG Vertretung, Arnold-Zweig-Straße 11, Berlin, 1100, Tel. 4724185

Ausstellung über CAD- und Bürotechnik

Vom 18. bis 19. November vergangenen Jahres zeigte die britische Firma RANK XEROX (RX), die ein Büro im Internationalen Handelszentrum in Berlin unterhält, im Berliner Hotel Metropol eine Auswahl ihrer Produkte. Neben Kopierern für Klein-, Mittel- und Großformat, Telekopierern und Schreibmaschinen stellte RX sogenannte Systemprodukte – vollständige Lösungen (Hard- und Software) für eine bestimmte Aufgabenstellung – vor. RX galt international lange Zeit als führender Hersteller von Kopiergeräten. Durch die immer stärker werdende Konkurrenz, vor allem von japanischen Kopiergeräteherstellern (z. B. Toshiba), „besinnt" man sich bei RX auf die ebenfalls langjährigen Erfahrungen im EDV-Bereich. So werden in ihren Leistungsparametern abgestufte Konfigurationen für CAD-Aufgaben (Bild 1) angeboten. Im Mittelpunkt steht dabei ein AT-kompatibler Mikrocomputer (RX OPat, RX AT Plus oder RX AT Compact). Die angebotenen PCs verfügen über einen 80286-μP - AT Plus und AT Compact können zusätzlich mit dem 80287 ausgerüstet werden. Je nach Ausführung sind 4 bis 6 AT-kompatible und 1 oder 2 XT-kompatible Steckplätze, Floppy-Disk bis zu 1,44 MByte, Festplatte bis zu 72 MByte und ein Streamerlaufwerk mit 60 MByte Kapazität vorhanden. Konfiguriert werden die PC wahlweise mit Plotter (schwarzweiß oder farbig), hochauflösendem Farbgrafikbildschirm, Digitalisiergerät, Drucker und dem Laserdrucker 4045. Der Laserdrucker (Bild 2), mit dem gleichzeitig kopiert werden kann, nutzt einen μP 80186 mit 512 KByte RAM, erweiterbar auf 2 MByte. An Schnittstellen sind RS 232 C und Centronics verfügbar. Die Papiergröße kann bis zu 21,6 cm × 35,5 cm bei 70–90 g/m^2 betragen. Der RX 4045 verursacht während des Betriebes einen Geräuschpegel von 50 db und im Bereitschaftszustand 40 db. Für den Druck gibt es eine Vielfalt von Schrifttypen (Fonts). Die Fonts variieren in der Größe von 6 bis 24 Punkten und sind mit festen oder proportionalen Schriftweiten erhältlich. Maximal lassen sich 22 Schrifttypen/-größen auf einer Seite verwenden. Auch Firmenzeichen und Unterschriften können (durch den Hersteller) auf Kassette gespeichert werden. Kyrillische Schrifttypen sind ebenfalls ladbar. Über die Centronics-Schnittstelle waren in der Ausstellung ein A7150 von Robotron und ein Laserdrucker 4045 verbunden (Bild 3). Auf dem A7150 erstellte Texte können mittels 4045 ausgegeben werden. Durch Nutzung weiterer Software soll das System als CAD-Station zur Verarbeitung und Ausgabe hochauflösender Grafik (300×300 Punkte je Quadratzoll) nutzbar sein. *MP*

Kosmos12
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890-+=/*()[]{}< >@#%!&?"':;,.

Classic12
ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ
abcdefghijklmnopqrstuvwxyzäöü
1234567890-+ =/*()[]{}< >@#%!&?"':;,.

SPOKESMAN10
ABCDEFGHIJKLMNOPQRSTUVWXYZ
ABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890-+=/*()[]{}<>@#%!&?"':;,.

2-MB-Floppy

Das BRD-Unternehmen TDK Electronics Europe GmbH, Ratingen bietet unter der Bezeichnung MF-2 Disketten an, die bei einem Durchmesser von 3,5 Zoll eine unformatierte Speicherkapazität von 2 Megabyte haben sollen. Diese Disketten werden nach Angaben des Unternehmens nach einem neuen, von TDK entwickelten Verfahren hergestellt, bei dem mittels eines Elektronenstrahls („Electron Beam Cured Binder"-Verfahren) eine wesentlich haltbarere Struktur der feinen Magnetbeschichtung aus super-feinkörnigen Avilyn-Partikeln auf getempertem Trägermaterial erreicht werde. Selbst nach 20 Millionen Durchläufen je Spur, gemessen bei 60 Grad Celsius, bringe diese Diskette noch volle Datensicherheit und bestmögliche Schreib- und Leseeigenschaften. Mit dem Elektronenstrahl-Verfahren könnten die Magnetpartikel besonders gleichmäßig, flach und dicht auf die Oberfläche der Disketten aufgebracht werden.

aus Blick durch die Wirtschaft

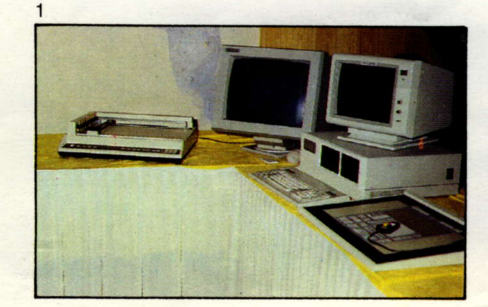
Speicherzelle mit „Bloch-Linie"

Hitachi hat in Zusammenarbeit der Universität Kyushu die Funktionsfähigkeit von Bloch-Linien-Speicherzellen praktisch nachgewiesen. Es handelt sich um einen magnetischen Schaltkreis, der Memory-ICs mit extrem hoher Zellendichte ermöglicht. Im Experiment wurde eine 5-Bit-Bloch-Speicherzelle auf einer Chipfläche von 200×400 μm aufgebaut. Der Erfolg des Experimentes eröffnet Möglichkeiten von Speicher-ICs mit einer Speicherdichte von maximal 16 Bit/μm^2. Hitachi möchte mit der Produktion der ersten Generation von 256-MBit-Bloch-Line-Memories schon in fünf Jahren beginnen. Eine Bloch-Linie ist eine zweidimensionale magnetische Region im Zusammenhang mit der „Bloch-Wand", die in einem ferromagnetischen Medium die Grenze zwischen zwei Weißschen Bezirken darstellt. In einem dünnen magnetischen Film entstehen genauso Zonen mit verschiedenen Magnetisierungsrichtungen, die von einer Wand getrennt werden. Die magnetische Region im rechten Winkel zu der Wand wird Bloch-Linie genannt, nach dem Physiker, der das Phänomen erforscht hat. Prof. Konishi von der Universität Kyushu hat nun vorgeschlagen, die Mikrostruktur entlang einer Bloch-Linie (mit Geometrien von 0,5 bis 1 μm) als digitale Speichermechanismen zu verwenden. Hitachi, NEC und mehrere Universitäten haben intensiv diese Möglichkeit erforscht. Der große Anreiz besteht darin, daß dadurch nichtflüchtige, statische RAMS realisiert werden können, die als monolithische ICs ohne bewegliche Teile die Kapazität von Floppy-Disks erreichen. Der Erfolg des Experiments wurde durch Verbesserungen der Transfer- und Schreiboperationen ermöglicht bzw. durch eine bisher noch unbekannte Auslesestruktur für die Bloch-Linie.

aus Elektronikschau 4/87

Fotos (4): Paszkowsky

1



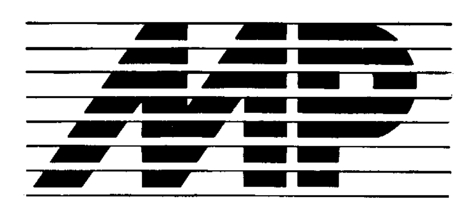
2



3



Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 3 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel.: 2870371); Sekretariat Tel.: 2870381

Gestaltung Christina Bauer

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

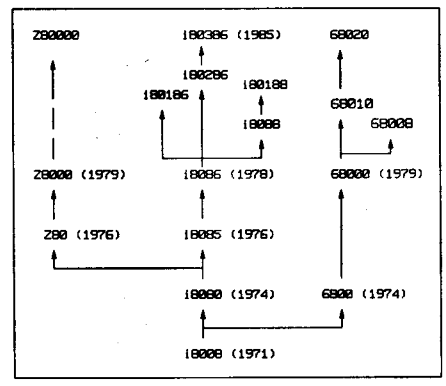
Redaktionsschluß: 12. Januar 1988

AN (EDV) 49837

Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tísku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Hà Nội; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010, und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



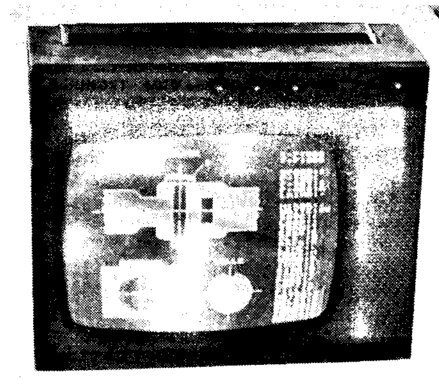
Seite 68

Block (KByte)	Blockshiftfaktor	Blockmask
1	3	07H
2	4	0FH
4	5	1FH
8	6	3FH
16	7	7FH

Der Wert der "Extentmask" ergibt sich aus der Blockgröße und der Speicherkapazität der Disk:

Block (KByte)	Extentmask bei Disk<256K	Disk>256K
1	0	geht nicht
2	1	0
4	3	1
8	7	3
16	15	7

Seite 74



Seite 90

Vorschau

In MP 4/88 haben wir für Sie u. a. Beiträge zu folgenden Themen vorgesehen:
- *Der Floppy-Disk-Controller U 8272D und sein Einsatz*
- *Schnelle Analog-/Digital-Wandlung und Sampling für 8/16-Bit-Computer*
- *Bauanleitung für einen Joystickmodul zum KC 85/3*

Inhalt

Videosteuerung VIS 3 mit GDC U82720D (Teil 1)

Dr. Wilfried Quednow
Akademie der Wissenschaften der DDR, Zentrum für wissenschaftlichen Gerätebau Berlin

Dem vielfachen Wunsch vieler Anwender der VIS1 und VIS2A Rechnung tragend, stellen wir in zwei aufeinanderfolgenden Beiträgen die Hard- und Software der neuentwikkelten Videosteuerung VIS3 vor. Wir möchten damit nicht nur eine Information über die Leistungsparameter und Bezugsmöglichkeiten dieser Baugruppe geben, sondern mit dem unkonventionellen Einsatz des Grafik-Display-Controllers GDC U82720D auf der VIS3 auch die Flexibilität und Leistungsfähigkeit dieses Schaltkreises zeigen. Unser Dank gilt dem Ingenieurbetrieb für Anwendung der Mikroelektronik Erfurt im KME und dem ZEG Mittweida, die unsere Arbeit maßgeblich unterstützt haben und durch ihre Beteiligung an der Fertigung und dem Vertrieb bestückter und unbestückter Leiterplatten mitzu einer breiten Praxiswirksamkeit beitragen werden.

Vorgängertypen

Im Zentrum für wissenschaftlichen Gerätebau werden auf dem Gebiet der Bildschirmgrafik bereits traditionell Hard- und Softwareentwicklungen durchgeführt, die in der DDR auf der Basis von Nachnutzungen eine große Verbreitung gefunden haben. Dazu gehören die beiden K1520-kompatiblen Steckeinheiten
– Videospeicher VIS1 /1/, /2/ mit einer Bildspeichergröße von 256 ∗ 265 ∗ 1 Bit und einem Anschluß für S/W-Fernsehgeräte
– Videospeicher VIS2A /3/, /4/ mit einer Bildspeichergröße von 512 ∗ 256 ∗ 1 Bit und einem Anschluß für die vom VEB Kombinat Robotron hergestellten monochromen Monitore MON K7221 bzw. MON K7222
sowie die betriebssystemunabhängigen grafischen Unterprogrammbibliotheken
– Grafikgrundsoftware GGS 180 zur Bedienung des Videospeichers VIS1
– Grafikgrundroutinen GGR 183 zur Bedienung des Videospeichers VIS2A
und die Anwenderprogramme
– Grafiksoftware, kurvenorientiert GSK 182 zur Darstellung von Meßkurven mit dem Videospeicher VIS1
– Grafiksoftware, kurvenorientiert GSK 183 zur Darstellung von Meßkurven mit dem Videospeicher VIS2A.

Dr.-Ing. Wilfried Quednow (42) studierte von 1964 bis 1970 an der Technischen Universität Dresden, Fachgebiet Nachrichten- und Informationstechnik. Nach seinem Diplomabschluß auf diesem Fachgebiet setzte er seine Tätigkeit an dieser Einrichtung als wissenschaftlicher Assistent fort und erlangte 1975 die Promotion A mit einer Arbeit über Computerteilnehmersysteme. Seit 1975 ist er im Zentrum für wissenschaftlichen Gerätebau der AdW der DDR tätig, wo er auf dem Gebiet der Rechentechnik und ihrer Anwendung im wissenschaftlichen Gerätebau arbeitet. Sein Spezialgebiet ist die Entwicklung von Bildschirmsteuerungen für Mikrorechner, die insbesondere auf die Belange der Meßtechnik zugeschnitten sind.

Auf Grund des großen Bedarfs an leistungsfähiger Bildschirmtechnik einerseits und des Angebotes neuer, dafür geeigneter hochintegrierter Schaltkreise andererseits erschien es uns sinnvoll, diese Serie fortzusetzen. Nachdem nunmehr die Hard- und Softwareentwicklungsarbeiten dazu abgeschlossen sind, sollen in diesem Beitrag die neuentwikkelte K1520-kompatible Steckeinheit Videosteuerung VIS3 /5/ und in einem weiteren Beitrag die zugehörigen Softwarekomponenten Farbgrafikroutinen FGR 186 und VIS3-Grafik-Library VGLIB vorgestellt werden.

Leistungsparameter

Die Videosteuerung VIS3 ist wie ihre Vorgängertypen eine vollgrafische Bildschirmsteuerung, die auf einer Steckeinheit im K1520-Standard realisiert ist. Ihre wichtigsten technischen Daten sind in Tafel 1 zusammengestellt.
Im Unterschied zur VIS1 und VIS2A verfügt sie über einen wesentlich größeren Bildschirmwiederholspeicher und einen erheblich erweiterten Funktionsumfang. Beides

Tafel 1 Technische Daten der VIS3

Bildvergrößerung (Zooming)	1 bis 16 (Zoomfaktor wirkt horizontal und vertikal gleichzeitig)
Zahl der Bildfenster	2
Bildfensterverschiebung (Panning)	
vertikal	im Zeilenraster
horizontal	im Raster von 4 Punkten
Palettenregister (Look up Table)	
Größe	16 ∗ 4 Bit
Zugriff	schreib-/lesbar (mit/ohne WAIT-Steuerung)
Rechnerinterface	K1520-Systembus, TGL 37271/01
Monitor	MON 7226.X0 /9/ (MON 7221, MON 7222 sowie S/W- und Farbfernsehempfänger anpaßbar)
Lichtstift	5366 – 20 (ZfK Rossendorf)
Stromversorgung	5 V ±5 %/0,9 A typ.
Steckverbinder	
Rechneranschluß	58pol. indirekt, TGL 29331/03
Lichtstiftanschluß	10pol. indirekt, TGL 29331/04
Monitoranschluß	26pol. indirekt, TGL 29331/04[2)]
	4pol. geschirmt, TGL 29331/05[2)]
	6pol. geschirmt, TGL 29331/05[2)]
Leiterkartenformat	215 mm ∗ 170 mm (EGS)

[1)] Vorzugsformat
[2)] alternative Bestückungsvarianten

wurde in erster Linie durch den Einsatz neuer und leistungsfähiger Schaltkreise, wie des GDC U82720 DC04 /6/ und des dynamischen 64-KBit-Speichers U2164 D25 /7/ erreicht. Im Bildwiederholspeicher können jetzt je Bildpunkt 4 Bit abgespeichert werden, wodurch die Ansteuerung eines Farbmonitors ermöglicht wird. Was den Leistungsumfang betrifft, wirken sich hier insbesondere die Möglichkeiten des GDC sehr vorteilhaft aus. Alle GDC-Kommandos nach /8/ können, soweit sie im Grafikmode zugelassen sind, benutzt werden. Im einzelnen stehen folgende programmierbare Funktionen zur Verfügung:
– Programmierung des Bildsynchronsignals (BSYN) einschließlich der Austastzeiten ermöglicht Anpassung an beliebige Monitore
– Umschaltung von Master-Betrieb (VSYNC-Erzeugung intern) auf Slave-Betrieb (VSYNC-Erzeugung extern) bei Kaskadierung
– programmgesteuerte An- und Abschaltung der Monitorsignale RT, BL, GR, INTS und BSYN (nach Hardware-RESET ist der Monitor grundsätzlich abgeschaltet)
– Auswahl, ob RMW-Zyklen nur während der Austastzeiten für Bild- und Zeilenrücklauf erlaubt sind oder ständig (Flash-Mode)
– Programmierung der Bildwiederholspeichergröße durch Eingabe der Zeilenlänge (PITCH)
– Programmierung der Größe des auf dem Bildschirm abgebildeten Bildfeldes und Aufteilung in ein oder zwei übereinander angeordnete Bildfenster (Windowing)
– voneinander unabhängige Verschiebung (Panning) der in den Bildfenstern dargestellten Bildausschnitte in horizontaler und vertikaler Richtung
– Vergrößerung (Zooming) des dargestellten Bildes durch gleichzeitige Aufweitung der Bildpunkte in vertikaler und horizontaler Richtung
– Lesen (RDAT) und Schreiben (WDAT) des Bildwiederholspeichers
– Lesen (DMAR) und Schreiben (DMAW) des Bildwiederholspeichers im DMA-Betrieb
– Zeichnen von Linien, Rechtecken und Kreisbögen mittels GDC-Zeichnungsprozessor
– Füllen von Flächen sowie Textzeichenausgabe mittels GDC-Zeichnungsprozessor
– Setzen eines grafischen, nicht sichtbaren Cursors (CURS) und Abfrage der aktuellen Cursorposition (CURD)
– Lesen der Lichtstiftposition (LPRD)
– Lesen und Schreiben der im Palettenregister (Look up Table) abgespeicherten Farbtabelle mit oder ohne WAIT-Generierung.

Funktionsweise

Die Blockschaltung der Videosteuerung VIS3 ist in Bild 1 dargestellt. Wesentlich für das Verständnis ihrer Funktion ist das Zusammenspiel zwischen Grafik-Display-Controller (GDC) und Bildwiederholspeicher (BWS). Daran beteiligt sind, wie noch zu erläutern ist, das Write-Enable-Register (WRITEN-REG) und das Zoomregister (ZOOM-REG). Die Schaltungsteile Adreßdekoder (DEC), Datenbustreiber (DB-DRIV), Adreßbustreiber (AB-DRIV) und Paletteneingaberegister (PAL-INREG) dienen der Anpassung an den K1520-Systembus und sind in ihrer Funktion trivial. Sie sollen deshalb hier nicht weiter erläutert werden.

Grundfunktion

Der Datenaustausch zwischen GDC und BWS erfolgt ausschließlich über den gemultiplexten Adreß-/Datenbus des GDC, der BSW-seitig in einen Datenein- und Datenausgabebus sowie in einen Adreßbus aufgefächert wird. Das geschieht mit Hilfe des Adreßmultiplexers (ADR-MUX) und des Dateneingaberegisters (BWS-INREG). Ein Datenaustausch zwischen GDC und BWS findet immer dann statt, wenn der GDC einen Lese- oder Schreibzyklus (RMW-Zyklus) ausführt, das heißt im GDC ein Lese- oder Schreibkommando (z. B. RDAT, WDAT) in Bearbeitung ist. Speicherzyklen ohne Datenaustausch zwischen GDC und BWS werden aus-

geführt, um den Inhalt des BWS ständig auf dem Bildschirm des Monitors darstellen zu können. Bei diesen sogenannten Displayzyklen liefert der GDC nur noch die fortlaufenden Adressen zum Auslesen der Speicherzellen des BWS, während ihr Inhalt, oder genauer gesagt das ausgelesene Displaywort, an die Videoschieberegister VSR übergeben wird. Diese erzeugen daraus durch Serialisierung die Videosignale, die anschließend vom Palettenregister (PAL-REG) manipuliert werden und nach Passieren des Videopuffers (VPU) und entsprechender Treiberstufen schließlich zum Monitor gelangen.

Bildwiederholspeicher

Eine Besonderheit auf der VIS3 ist die gewählte Aufteilung des BWS in 4 Bildebenen zur Realisierung von 4 Bit je Bildpunkt. In Bild 2 sind das für den GDC allgemein übliche Prinzip der Speicherbankaufteilung (Bild 2a) und die hier angewendete Aufteilung des Displaywortes in Segmente (Bild 2b) gegenübergestellt. Während der GDC bei der Arbeit mit Speicherbanken nur jede Bildebene einzeln und nacheinander ansprechen kann, also beim Zugriff auf einen Bildpunkt immer mehrere RMW-Zyklen ausführen muß, ist bei der Segmentierung der gleichzeitige Zugriff zu allen Bildebenen innerhalb eines RMW-Zyklus möglich. Das bringt einen erheblichen Geschwindigkeitsgewinn beim Bildaufbau mit sich und bietet darüber hinaus auch den Vorteil, bereits mit 16 Speicherschaltkreisen der Organisation 64 K ∗ 1 Bit vier Bildebenen realisieren zu können. Dabei wird zwar prinzipiell die maximal erreichbare Videotaktfrequenz eingeschränkt, was hier aber nicht weiter von Bedeutung ist. Auswirkungen hat es aber auf die GDC-Programmierung und die Schaltungskonzeption.
Hinsichtlich der Programmierung ist zu beachten, daß mit dem GDC-Cursorkommando sowohl die Adressierung des Displaywortes als auch die Auswahl des entsprechenden Bits innerhalb des Displaywortes erfolgt. Letzteres geschieht mit Hilfe des GDC-Maskenregisters, in dem das betreffende Maskenbit automatisch auf 1 gesetzt wird. Für die Wortsegmentierung müssen aber nicht nur das eine, sondern gleichzeitig alle 4 zu einem Bildpunkt gehörenden Bits modifiziert werden können. Deshalb ist hier zusätzlich nach dem GDC-Cursorkommando immer noch ein Maskenwort auszugeben, mit dem diese Bits nachträglich gesetzt werden. Wird dieser Grundsatz eingehalten, können trotz Segmentierung des BWS auch weiterhin alle GDC-Kommandos einschließlich des GDC-Zeichnungsprozessors im vollen Umfang genutzt werden.

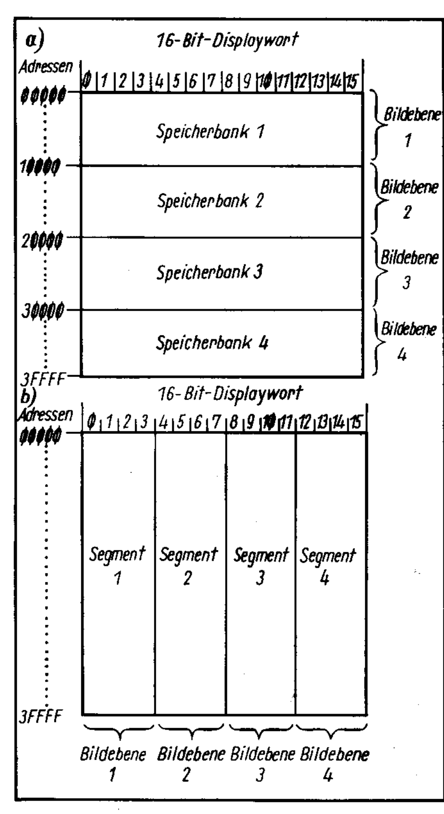


Bild 2 Aufteilung des BWS
a) nach Speicherbanken
b) nach Wortsegmenten

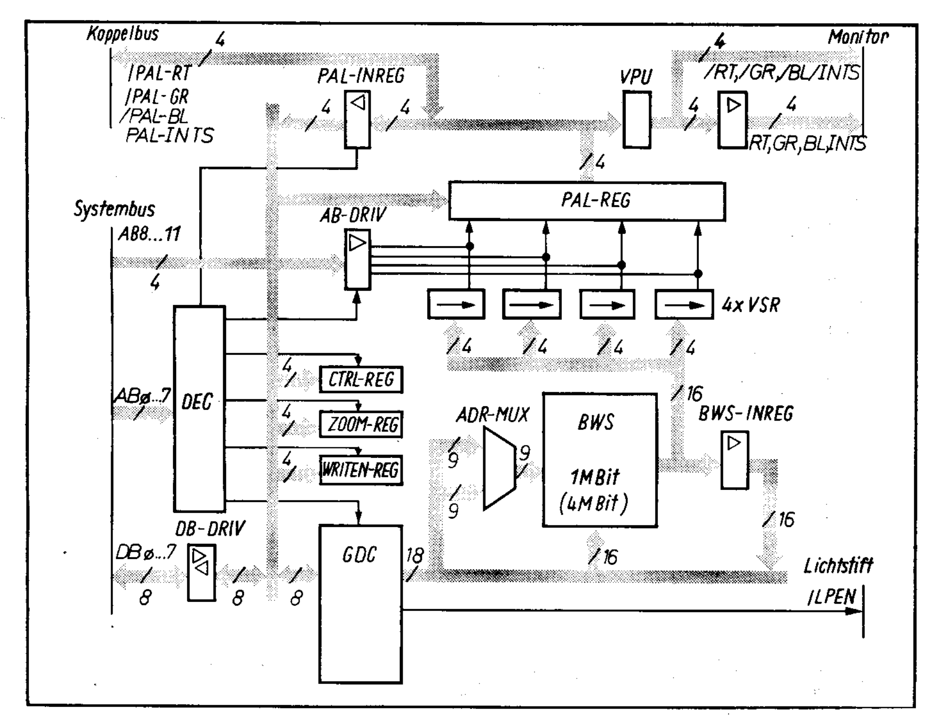


Bild 1 Blockschaltung VIS3

Schaltungstechnisch verlangt das Prinzip der Wortsegmentierung die Möglichkeit, Segmente bzw. Bildebenen während der Ausführung von RMW-Zyklen einzeln sperren und freigeben zu können. Das ist erforderlich, weil sonst auf Grund der oben beschriebenen Belegung des GDC-Maskenregisters Schreiboperationen des GDC immer in allen Bildebenen gleichzeitig ausgeführt werden würden. Auf der VIS3 wird diese Funktion vom Register WRITEN-REG übernommen. Es kann von der ZRE des K1520 geladen werden und maskiert jedes der 4 Schreibsignale des BWS entsprechend den 4 Segmenten des Displaywortes.

Palettenregister (Look up Table)

Das Palettenregister PAL-REG hat die üblichen Funktionen zu erfüllen. Es enthält eine Farbtabelle, die jeder Bitkombination eines Bildpunktes eine entsprechende Farbe zuordnet, und es kann von der ZRE des K1520 gelesen und geschrieben werden. Das Zugriffsregime, das heißt, ob das PAL-REG ständig oder nur während der Zeilenlücken ansprechbar ist, kann dabei über das CTRL-REG eingestellt werden. Im letzten Fall wird so lange ein WAIT-Signal an die ZRE des K1520 gesendet, bis eine Zeilenlücke erreicht wird.

Bildvergrößerung

Die hardwaregesteuerte Bildvergrößerung wird im wesentlichen durch den GDC selbst und einen programmierbaren Taktteiler realisiert. Vom Taktteiler werden der Übernahmetakt und der Schiebetakt der VSR abgeleitet. Seine Programmierung erfolgt durch Ausgabe des gewünschten Teilerverhältnisses an das Register ZOOM-REG. Erhält der GDC das Kommando ZOOM, führt er die Vergrößerung des Bildes in der Weise aus, daß er jede Displayzeile mehrfach wiederholt (vertikale Bilddehnung) und die Displayzyklen zeitlich verlängert (horizontale Bilddehnung). Die Verlängerung der Displayzyklen allein führt dabei aber noch nicht zur gewünschten horizontalen Bilddehnung. Sie wird erst wirksam, wenn damit auch gleichzeitig der Übernahmetakt und der Schiebetakt der VSR herabgesetzt werden. Das erfolgt, indem anschließend einfach ein neues Teilerverhältnis an das ZOOM-REG ausgegeben wird.

DMA-Betrieb

Beim DMA-Betrieb kann der GDC den BWS blockweise lesen und schreiben. Diese Fähigkeit des GDC ist auf der VIS3 ebenfalls nutzbar, allerdings ohne Unterstützung durch einen DMA-Controller. Die DMA-Zugriffe unterliegen hier voll der Programmsteuerung der ZRE des K1520, das heißt, die GDC-Signale DRQ und DACK sind softwaremäßig zu bedienen. Dabei wird das GDC-Signal DRQ über das Register CTRL-REG geschaltet.

Anwendung und Bezugsmöglichkeiten

Die Videosteuerung VIS3 ist insbesondere für die Einsatzfälle geeignet, wo ein bestehendes 8-Bit-Mikrorechnersystem durch ein grafikfähiges und farbtüchtiges Bildschirmausgabesystem ergänzt werden soll. Dabei ist es zweckmäßig, den in der Regel vorhandenen Alpha-Bildschirm beizubehalten, um so die Betriebs- und Anwendersoftware auch nach der Umrüstung noch unverändert nutzen zu können. In diesem Sinne ist die VIS3 in folgenden Mikrorechnersystemen verwendbar:

– Bürocomputer BC A 5120/5130
– Personalcomputer PC 1715 (Anschluß an internen Steckverbinder X1)
– Mikrorechnerfamilie MC 80
– Mikroprozessorsystem MPS 4944 (Anschluß über Adapter ADP1)
– anwenderspezifische K1520-kompatible Mikrorechnersysteme.
Natürlich besteht auch die Möglichkeit, auf den separaten Alpha-Bildschirm zu verzichten und sämtliche Bildschirmausgaben über den Farbmonitor zu leiten. Dies ist jedoch im allgemeinen mit sehr hohem Softwareänderungsaufwand verbunden und sicher nur in Ausnahmefällen sinnvoll.

Die Videosteuerung VIS3 wird vom VEB Mikroelektronik „Karl Marx" Erfurt, Ingenieurbetrieb für die Anwendung der Mikroelektronik, Struktur CI, Rudolfstr. 47, Erfurt, 5010, als nachnutzbare Lösung angeboten. Die Nachnutzung erfolgt über Nachnutzungsverträge und beinhaltet die Lieferung unbestückter Leiterplatten sowie des kompletten Unterlagensatzes der VIS3.
Begrenzte Bezugsmöglichkeiten bestehen auch beim Zentrum für elektronischen Gerätebau, Platz der DSF 17, Mittweida, 9250, und beim
Zentrum für wissenschaftlichen Gerätebau (ZWG), Abt. Absatz, Rudower Chaussee 6, Berlin, 1199.
Diese Kapazitäten sind jedoch vorrangig zur Abdeckung des Eigenbedarfs im Bereich des Ministeriums für Hoch- und Fachschulwesen bzw. der AdW und AdL der DDR bestimmt.
Der Softwarevertrieb wird ausschließlich vom ZWG im Rahmen von Nachnutzungsverträgen durchgeführt.

wird fortgesetzt

Literatur
/1/ Quednow, W.: Vollgrafisches Display für Mikrorechner. Feingerätetechnik, Berlin 31 (1982) 12, S. 573
/2/ Grafikdisplay GDM 180 und GDM 180/1. Geräte-Information des ZWG, Berlin 1983
/3/ Puder, J.: Grafikdisplay GDM 183 und GDM 183/1. Feingerätetechnik, Berlin 33 (1984) 8, S. 374
/4/ Grafikdisplay GDM 183 und GDM 183/1. Geräte-Information des ZWG, Berlin 1984
/5/ Videosteuerung VIS3, Geräte-Information des ZWG, In Vorbereitung
/6/ Bankel, M.; Brückner, P.; Wolf, R.: Grafik-Interface mit dem U 82720. Mikroprozessortechnik, Berlin 1 (1987) 4, S. 99–103
/7/ De Paly, T.; Bürger, B.; Schnieck, H.-G.: 64-KBit-DRAM mit wahlfreiem Zugriff U 2164 C. Radio, Ferns., Elektron., Berlin 36 (1987) 4, S. 241–242 und 36 (1987) 5, S. 309–310
/8/ GDC U 8270 – Technische Beschreibung. VEB Mikroelektronik „Karl Marx" Erfurt, 1987
/9/ Dietze, K.-H.; Pilz, P.: Der Farbmonitor K7226 und seine Anschlußsteuerung. Neue Technik im Büro, Berlin 29 (1985) 2, S. 33–35

KONTAKT
Akademie der Wissenschaften der DDR, Zentrum für wissenschaftlichen Gerätebau, Rudower Chaussee 6, Berlin, 1199; Tel. 6743381

Moderne Mikrorechnersysteme

Prof. Dr. Peter Neubert, Ralph Willem, Karsten Künne
Technische Universität Dresden, Sektion Informationstechnik

In der Mikrorechentechnik haben neben den Speicherschaltkreisen vor allem die Mikroprozessoren eine gewaltige Weiterentwicklung erfahren. Neben der Erhöhung der Datenbreite und der Taktfrequenz bei Universalprozessoren wurden leistungsfähige Spezialprozessoren entwickelt, so daß zur Zeit eine große Anzahl von Prozessortypen verwendet wird. Dem wollen wir mit einer Übersichtsbeitragsfolge über die 16- und 32-Bit-Mikroprozessoren der i8086-Familie Rechnung tragen; dieser werden sich ausführliche Beiträge über den i8086 von Intel und den MC6800 von Motorola anschließen.
Wir meinen, daß wir damit unseren Lesern eine gute Ausgangsposition bei der Einarbeitung in diese neue Leistungsklasse in der Mikrorechentechnik bieten.

1. Einführung

In den vergangenen vier bis fünf Jahren hat sich auf dem Gebiet der Mikroprozessoren (MP) eine rasante Entwicklung vollzogen. Zur Zeit spricht man vom vollständigen Übergang von 8- zu 16-Bit-MP und dem immer breiteren Einsatz von 32-Bit-MP. Bild 1 stellt eine Art Stammbaum einiger ausgewählter MP dar /1/, /2/.
Dabei kann man den MP i8008 als „Vater" aller heutigen MP betrachten. Er stellte mit dem i4004 die ersten Versuche dar, eine aus der Prozeßrechentechnik bekannte Rechnerarchitektur „in Silizium zu gießen". Aus diesen entwickelten sich mehrere Prozessorlinien, von denen drei im obigen Bild auszugsweise dargestellt sind. 1976 entstanden die ersten 16- und Mitte der 80er Jahre die ersten 32-Bit-MP. Inhalt dieses Beitrages soll es sein, die 16- und 32-Bit-MP von Intel vorzustellen und die Geradlinigkeit der Entwicklung dieser Reihe aufzuzeigen.

Tafel 1 Beschaltung des MN/MX-Pins der CPU

MN/MX	Arbeitsweise
1	Minimummode
0	Maximummode

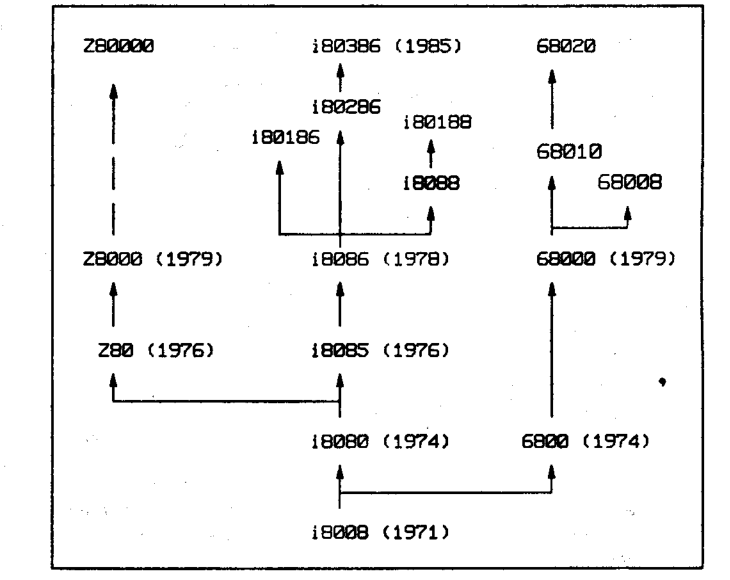


Bild 1 Entwicklung ausgewählter Mikroprozessoren

2. Das System MCS-86

2.1. Bestandteile einer funktionsfähigen Zentraleinheit

Das System MCS-86 gestattet den einfachen Entwurf von Mehrrechnersystemen. Es unterstützt zwei Typen von Prozessoren:
– unabhängige Prozessoren, welche ihren eigenen Befehlsstrom bearbeiten (E/A-Prozessor)
– Coprozessoren, welche ihren Befehlsstrom von einem anderen unabhängigen Prozessor erhalten.
Um eine effektive Anpassung an die gegebenen Anwendungsbedingungen zu erreichen, kann der 8086/8088 in zwei Arbeitsmodi betrieben werden. Für beide Modi gilt gemeinsam:
– Der direkt adressierbare Speicherraum beträgt 1 MByte.
– Der E/A-Adreßraum ist 64 KByte groß.
– Die Adreß- und Dateninformationen werden, auf Grund der begrenzten Pinzahl des Gehäuses der CPU, zeitmultiplex übertragen.
– Die Taktversorgung der CPU wird durch einen speziellen Taktgenerator realisiert. Dieser erzeugt den Takt mit einem Tastverhältnis von 2:1, synchronisiert das Resetsignal und generiert das Readysignal für die CPU.

Das jeweilige Arbeitsregime wird durch die Beschaltung des MN/MX-Pins der CPU festgelegt (Tafel 1).
Alle im folgenden für den 8086 gemachten Aussagen gelten sinngemäß auch für den 8088, wenn beachtet wird, daß der 8088 nur einen 8 Bit breiten externen Datenbus besitzt.

2.1.1. Der Minimummode

Soll die 8086-CPU in kleineren oder mittleren Systemen zur Anwendung kommen, so kann der Minimummode verwendet werden. In diesem Arbeitsmode generiert die CPU alle notwendigen Steuersignale selbst. Um die Zusammenarbeit mit z. B. einem DMA-Schaltkreis zu ermöglichen, ist ein einfaches Busübergabeprotokoll implementiert (HOLD,

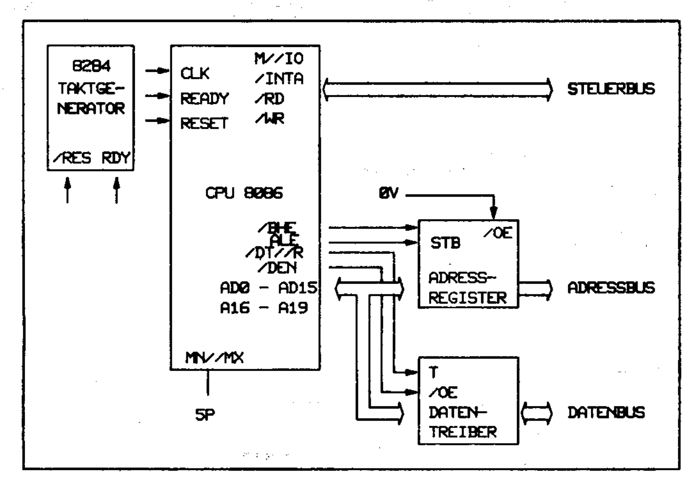




Bild 2 8086 im Minimummode

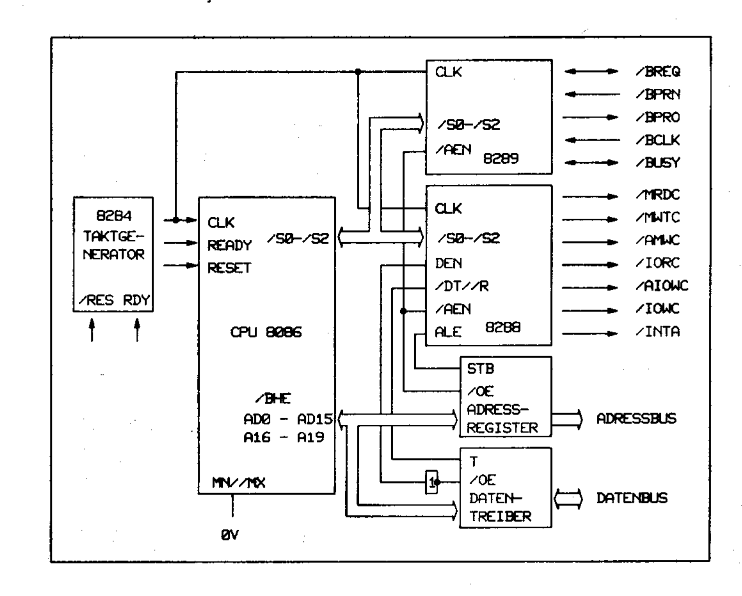


Bild 3 8086 im Maximummode

HOLDA). Die Anschaltung von Coprozessoren oder der Einsatz in Multimastersystemen ist damit nicht möglich. Bild 2 stellt eine im Minimummode arbeitende ZVE dar. Die Register und Datentreiber werden benötigt, um den zeitmultiplexen Adreß- und Datenbus zu trennen.

2.1.2. Der Maximummode

Die volle Leistungsfähigkeit, insbesondere in Mehrrechnersystemen, kann der 8086 nur im Maximummode erreichen. Auch der Einsatz von Coprozessoren ist nur im Maximummode möglich. Für diese zusätzlichen Möglichkeiten müssen von der CPU weitere Steuerfunktionen realisiert werden, wodurch sich eine Umbelegung von Pins notwendig macht. Durch den Einsatz von je einem speziellen Schaltkreis für die Busverwaltung und die Bussteuerung werden an der CPU weitere Steuerleitungen frei. Bild 3 zeigt den typischen Aufbau eines 8086 im Maximummode. Mit den /RT//GTO- und /RT//GT1-Leitungen wird die Kopplung von Coprozessoren ermöglicht.
Die Verwaltung des Zugriffs auf den Multibus realisiert der Busarbiter 8289. Die Steuerung der Busarbeit bei zugeteiltem Bus übernimmt der Buscontroller 8288. Die Funktion dieser beiden ICs soll in den folgenden Abschnitten näher erläutert werden.

2.1.2.1. Bussteuerschaltkreis 8288

Im Maximummodesystem übernimmt der 8288 alle notwendigen Steuerfunktionen für den Systembus sowie für die Steuerung der Adreß- und Datentreiber. Bild 4 zeigt eine Blockschaltung dieses IC.
Die CPU gibt drei Statussignale an beide Steuerbausteine aus. In diesen drei Signalen sind die durchzuführenden Zyklen nach Tafel 2 kodiert.
Entsprechend den empfangenen Statussignalen aktiviert der 8288 die notwendigen Lese-, Schreib- oder andere Kommandoleitungen. Des weiteren sind Steuerleitungen für die Adreßregister (ALE) und Datentreiber (DT//R, DEN) zum Demultiplexen des Adreß-/Datenbusses der CPU vorhanden. Ihre Wirkungsweise ist im Taktdiagramm nach Bild 5 dargestellt. Ein Buszyklus besteht mindestens aus vier Takten. Mit dem high-aktiven Signal ALE werden die Adressen in die Register eingeschrieben. Das Signal DEN öffnet die Datentreiber. Ihre Übertragungsrichtung legt das Signal DT//R fest. Der Buscontroller besitzt weiterhin Eingänge zu seiner eigenen Steuerung. Wichtig ist hier vor allem der Eingang /AEN, welcher die Mehrrechnerfähigkeit gestattet. Wird /AEN inaktiv, so gehen alle Kommandoausgänge in tri-state. Damit können mehrere 8288 an einem Bus arbeiten. Die /AEN-Erzeugung ist Aufgabe des Busverwalters 8289.

Tafel 2 Kodierung der Buszyklen in den Statussignalen

/S2	/S1	/S0	Buszyklus
0	0	0	Interruptannahmezyklus
0	0	1	E/A-Lesezyklus
0	1	0	E/A-Schreibzyklus
0	1	1	Halt
1	0	0	Befehlsholezyklus
1	0	1	Speicher-Lesezyklus
1	1	0	Speicher-Schreibzyklus
1	1	1	Kein Buszyklus

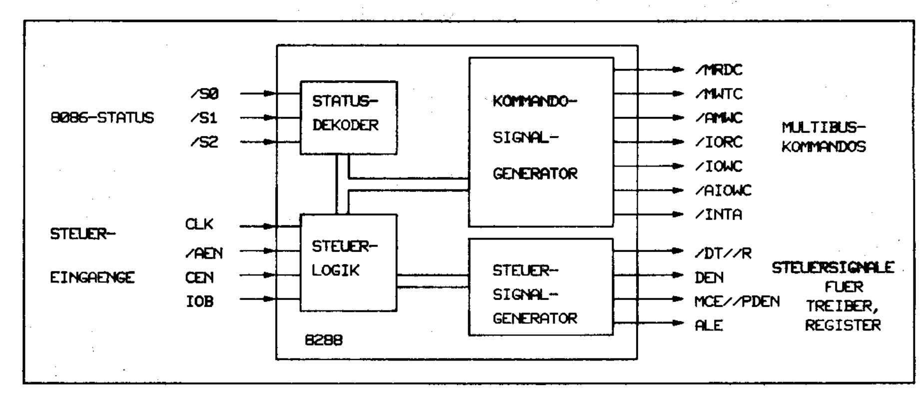
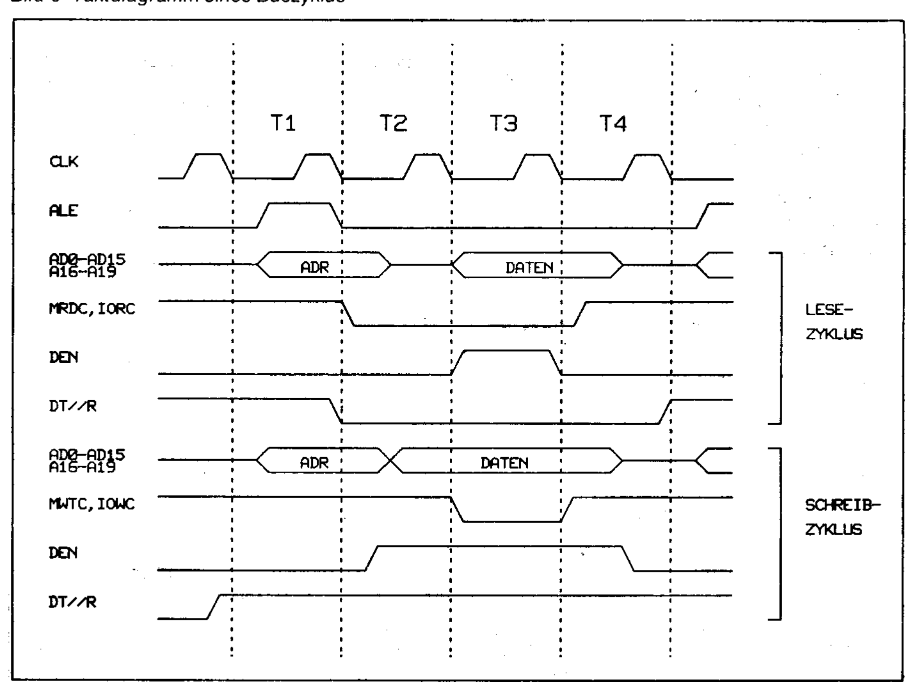
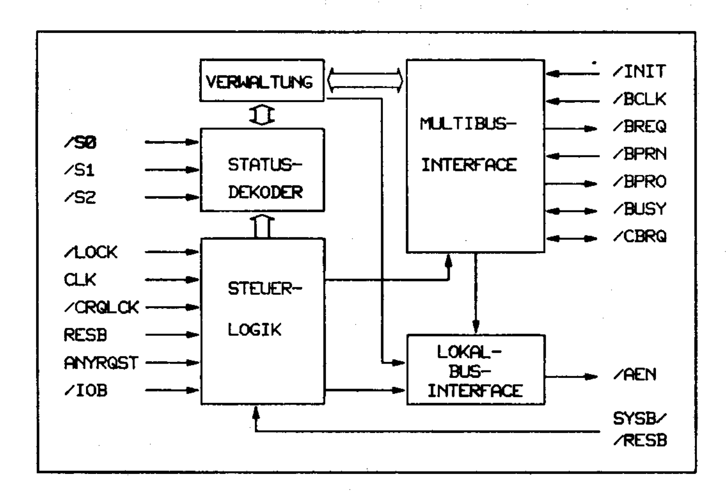


Bild 4 Blockschaltung des 8288

Bild 5 Taktdiagramm eines Buszyklus





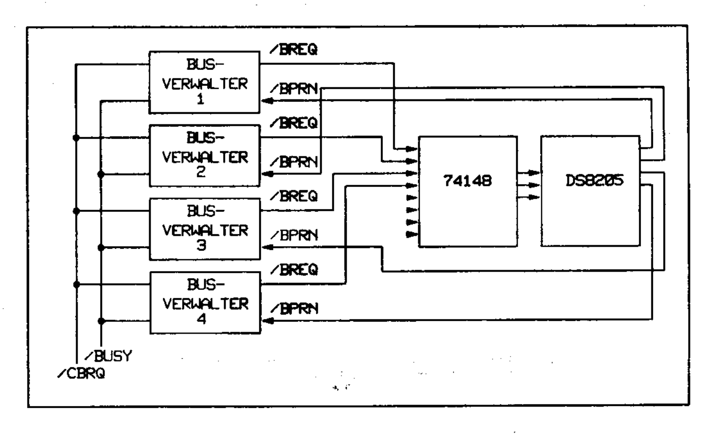


Bild 8 Parallele Prioritätsentscheidung bei Busvergabe

Bild 6 Blockschaltung des Busverwalters 8289

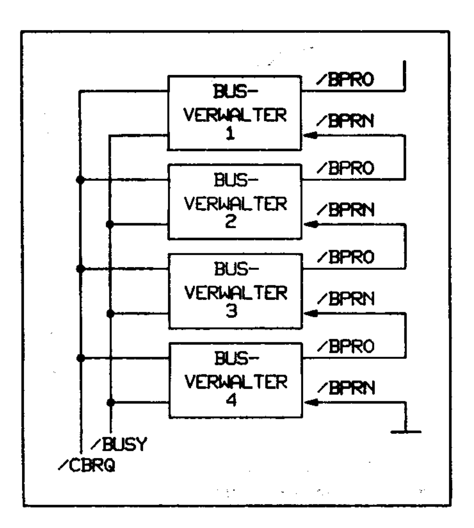


Bild 7 Serielle Prioritätsentscheidung bei der Busvergabe

2.1.2.2. Busverwalter 8289

Der Busverwalter realisiert gemeinsam mit dem 8288 die Kopplung von 8086/8088/8089-MP an den Multibus. Er steuert den Zugriff auf den gemeinsamen Bus. Verbietet er den Buszugriff, so führt die CPU Waitzyklen aus. Damit bleibt die Arbeit des Arbiters transparent für die CPU. Bild 6 zeigt eine Blockschaltung dieses IC.
Man kann vier Signalgruppen unterscheiden:
1. Multibus-Kommandosignale
Sie dienen der Steuerung der Busanforderung und -vergabe.
2. CPU-Statussignale
Sie legen den jeweiligen Zyklus fest und lösen, wenn notwendig, eine Busanforderung aus.
3. Systemsignale
Steuersignal für Buscontroller sowie Adreßregister
4. Busverwaltersteuerung
Signale, um den Busverwalter an die jeweilige Konfiguration anzupassen.

Für die Busvergabe selbst werden zwei Prioritätsmodi angewendet:
- serielle Prioritätskette (daisy chain)
- parallele Prioritätsentscheidung.

Die Bilder 7 und 8 stellen beide Varianten dar.
Die serielle Verkopplung der Arbiter benötigt den geringsten Aufwand. Hier ist allerdings die Anzahl der in einem System möglichen Busarbiter begrenzt. Bei 10 MHz Bustakt können maximal drei Arbiter in Reihe geschaltet werden. Sollen mehr als drei Master gekoppelt werden, so muß entweder der Bustakt verringert werden oder die parallele Verwaltungslogik zur Anwendung kommen.
Für beide Varianten gilt, daß immer der Rechner am Bus arbeiten kann, wo der /BPRN-Eingang des zugeordneten Arbiters aktiv ist. Das aktive Signal /BUSY zeigt an, daß der Bus genutzt wird.

2.1.3. Interruptsystem und programmierbarer Interruptcontroller 8259A

Das Unterbrechungssystem des 8086 ist einfach und dennoch sehr leistungsfähig. Es sind bis zu 256 verschiedene Unterbrechungen möglich. Alle Unterbrechungen sind vektororganisiert. Die Interrupttabelle beginnt auf der Adresse 0000:0000 und enthält für jede Unterbrechung zwei 16-Bit-Wörter, welche die Adresse der Behandlungsroutine darstellen. Bei einer Unterbrechung wird der CPU nur der Typ(Nummer) der Unterbrechung mitgeteilt. Durch Multiplikation des Typs mit 4 ermittelt die CPU den Zeiger in die Interrupttabelle. Die Interruptquellen können interner und externer Natur sein.
Interne Interrupts sind:

Division durch 0	Typ 0
Einzelschritt	Typ 1
Überlauf	Typ 4
Softwareinterrupt	Typ n

(durch die Anweisung INT n ausgelöst).
Externe Interrupts können sein:

NMI	Typ 3
maskierbare Interrupts	Typ entsprechend PIC-Programmierung.

Nur bei externen maskierbaren Unterbrechungen wird der Typ mittels eines Interruptannahmezyklus über den Datenbus übertragen. Alle anderen Unterbrechungen besitzen vordefinierte Typen.
Die Peripherieschaltkreise von Intel unterstützen keinen vektorisierten Interrupt. Da die CPU den Typ zur ordentlichen Behandlung benötigt, gibt es einen Interruptcontroller, welcher die Unterbrechungsanforderungen der peripheren Bauelemente annimmt und einen entsprechenden Typ zur CPU überträgt. Bild 9 zeigt das Blockschaltbild dieses Controllers.
Der PIC bedient 8 externe Interruptquellen. Ihre Priorität ist untereinander frei programmierbar. Durch das Maskenregister können einzelne Interrupteingänge maskiert werden. Die Steuerlogik realisiert das notwendige Signalspiel zur Unterbrechungsanmeldung an die CPU und die Übergabe des Interrupttyps. Über die Schreib-/Leselogik können der Baustein programmiert und die Statusregister gelesen werden.
Die Kaskadierungseinheit erlaubt die Kaskadierung von PICs. Damit können maximal 64 externe Interrupts behandelt werden. Zu diesem Zweck werden die Interruptanmeldungsleitungen der Slave-PIC mit je einem IR n Eingang des Master-PIC verbunden. Die Prioritätsentscheidung geschieht nun in zwei Stufen. In der ersten Ebene wird vom Master-PIC der Slave-PIC mit der höchsten Priorität ermittelt und dessen Adresse über CASO-CAS2 an die Slave-PICs ausgegeben. Der so angesprochene Slave-PIC sendet nun den Typ der höchstpriorisierten Interruptanforderung an die CPU.
Durch eine sehr freizügige Programmierung kann der PIC allen Systemerfordernissen angepaßt werden.

2.2. *Die CPU 8086*

Die CPU 8086 ist in einem 40-Pin-DIL-Gehäuse untergebracht. Sie besitzt einen zeitmultiplexen Adreß-/Datenbus.

2.2.1. Architektur des 8086

Bild 10 zeigt die Architektur des 8086 /3/. Man kann die CPU in zwei Einheiten teilen, in die Businterfaceeinheit (BIU) und die Befehlsausführungseinheit (EU).
In der BIU sind die Segmentregister, die Bussteuerlogik, eine 6 Byte lange (8088 nur 4 Byte) Befehlswarteschlange und der Befehlszähler (IP) untergebracht.
Die EU enthält die Universalregister, das Flagregister, die ALU und ihr Steuersystem. Beide Einheiten arbeiten asynchron. Benötigt die EU einen Zugriff zu E/A-Geräten oder Speichern, sendet sie eine Busanforderung zur BIU. Wenn zu diesem Zeitpunkt kein Buszyklus läuft, gibt die BIU dieser Anforderung statt und beginnt den Zyklus. Andernfalls führt die EU Wartetakte aus. Liegen keine EU-Busanforderungen vor, so füllt die BIU die 6 Byte lange Befehlswarteschlange auf. Aus dieser Warteschlange holt sich die EU die Befehlscodes; dafür werden von der EU keine Buszyklen angefordert.

2.2.2. Registersatz

Die CPU 8086 besitzt 14 16-Bit-Register. Sie lassen sich in folgende Gruppen einteilen:
- Datenregister
- Pointer- und Indexregister
- Segmentregister

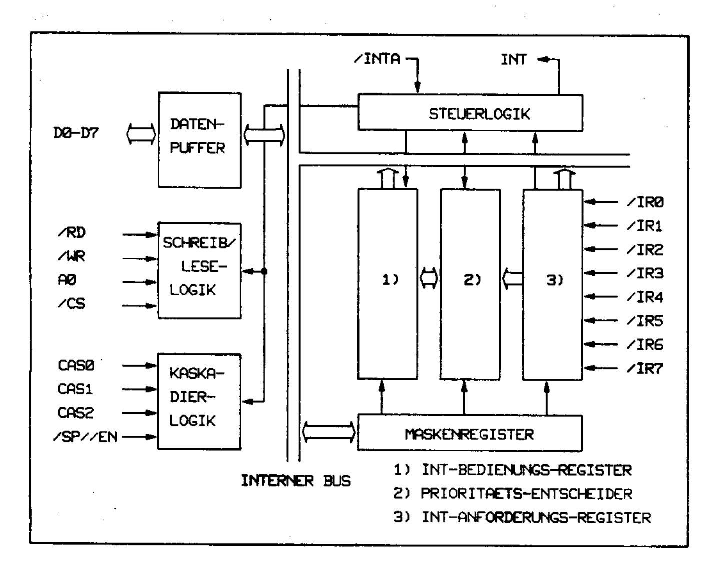


Bild 9 Blockschaltung des PIC 8259A

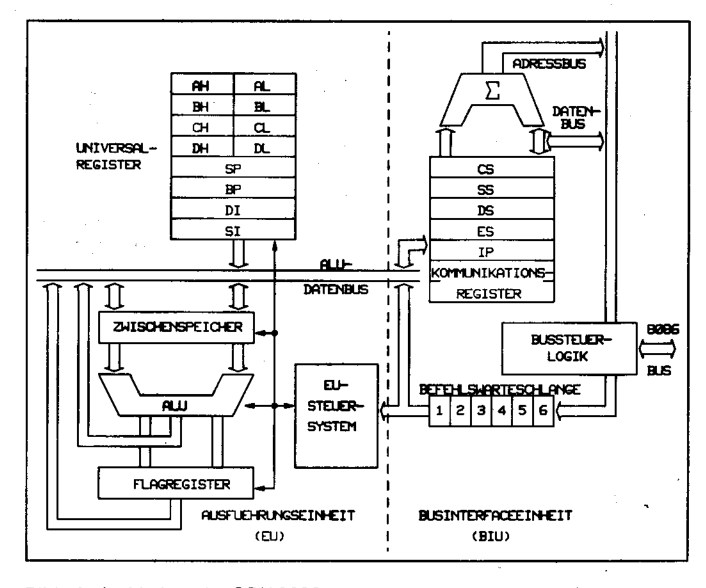


Bild 10 Architektur der CPU 8086

– Flagregister und Befehlszähler.
Die Datenregister können alle als 8-Bit-Register verwendet werden. Auf alle anderen Register kann nur in Wortbreite zugegriffen werden. Die Datenregister können in fast allen logischen und arithmetischen Operationen als Operandenspeicher dienen. Um eine kompaktere Kodierung zu erreichen, nutzen einige Befehle Daten-, Pointer- und Indexregister implizit.
– Register CX dient als Zählregister für Schleifen, Rotationen und Verschiebungen.
– Register DX kann eine 16-Bit-E/A-Adresse beinhalten.
– Register SP arbeitet als Stackpointer.
– Register SI und DI enthalten Quell- bzw. Zieladresse bei Kettenoperationen.

Der gesamte Speicherraum von 1 MByte ist physisch in zwei Bänke und logisch in maximal 64 KByte lange Segmente untergliedert. Die zwei physischen Speicherbänke besitzen jeweils eine Datenbreite von 8 Bit und enthalten die Speicherzellen nach gerader und ungerader Adresse getrennt. Bei Wortzugriffen wird mit beiden Bänken gleichzeitig, bei Bytezugriffen nur mit der angesprochenen Bank gearbeitet. Die Auswahl erfolgt durch die Adreßleitung AO und das Steuersignal /BHE (/BHE = 0 gibt an, daß Daten auf D8-D15 übertragen werden). Logisch kann die CPU zu jedem Zeitpunkt auf vier verschiedene Segmente zugreifen, deren Anfangsadressen in den vier Segmentregistern stehen.
Das CS-Register enthält immer den Anfang des aktuellen Kodesegments. Aus diesem Segment wird immer der Befehlsstrom entnommen.
Das SS-Register zeigt immer auf den Anfang des Stacksegmentes. Alle Kelleroperationen laufen in diesem Segment ab.
Das ES- und DS-Register zeigen auf den Anfang des Daten- bzw. Extrasegmentes. Diese Segmente enthalten normalerweise Variablen. Alle beide werden bei Kettenoperationen genutzt.
Der IP enthält den Offset im aktuellen Kodesegment. In ihm steht immer die Adresse des nächsten durch die BIU einzulesenden Befehls. Zwischen der Adresse des nächsten durch die EU auszuführenden Befehls und der Adresse des nächsten durch die BIU einzulesenden Befehls kann auf Grund der Befehlswarteschlange ein Unterschied sein. Deshalb wird der IP-Inhalt vor allen Operationen, die das Register IP speichern, korrigiert, und er enthält dann die Adresse des nächsten abzuarbeitenden Befehls.
Im Flagregister sind Steuer- und Statusbits untergebracht. Neben den Standardbits für Übertrag (CF), Vorzeichen (SF), Null (ZF), Parität (PF) und Überlauf (OF) sind noch weitere Informationen verfügbar.
AF – Hilfsübertrag:
Kennzeichnet den Übertrag von Bit 3 nach Bit 4 eines Bytes.
TF – Trap-Flag:
Wird dieses Bit gesetzt, so arbeitet die CPU im Einzelschrittbetrieb, das heißt, nach der Ausführung jedes Befehls wird ein Interrupt vom Typ 1 ausgelöst.
IF – Interruptflag:
Durch Rücksetzen dieses Bits können alle maskierten Interrupts verboten werden.
DF – Richtungsflag:
Dieses Richtungsflag bestimmt, ob die Inhalte des DI- und SI-Registers dekrementiert oder inkrementiert werden (Kettenoperationen). Ist DF = 1, so werden die Indexregister dekrementiert.

2.2.3. Adressierung

Eine Adreßangabe besteht beim MCS-86-System immer aus zwei Teilen:
1. Startadresse des Segments
2. Verschiebung innerhalb des Segments.
Die Startadresse der Segmente wird den Segmentregistern entnommen. Die Verschiebung, oder effektive Adresse (EA) genannt, wird mittels der unter 2.2.3.2. beschriebenen Adressierungsarten bereitgestellt.

2.2.3.1. Umwandlung der logischen in die physische Adresse

Die Startadresse des Segments und die EA sind 16-Bit-Wörter. Der direkt adressierbare Speicherraum erfordert eine 20-Bit-Adresse. Die CPU errechnet die physische Adresse nach folgender Vorschrift:
1. Inhalt des Segmentregisters um 4 Bit nach links verschieben, dazu rechts Nullen einschieben.
2. EA mit Nullen auf 20 Bit ergänzen (links).
3. Beide Größen addieren.
Aus dieser Vorschrift folgt, daß Segmente nur an einer durch 16 teilbaren Adresse beginnen können. Im allgemeinen wird folgende Darstellung benutzt:
Segmentadresse: effektive Adresse
Fehlt die Segmentangabe, so werden vordefinierte Standards verwendet.

2.2.3.2. Adressierungsarten

1. Speicheradressierung

– direkte Adressierung:
Die EA wird direkt dem Befehl entnommen.
– Register-indirekte Adressierung:
Die EA befindet sich in einem Basis- oder Indexregister.
– Basisadressierung:
Die EA ergibt sich aus der Summe des BP- oder BX-Registers mit einer Verschiebung.
– Indizierte Adressierung:
Wie bei der Basisadressierung, nur daß statt BP oder BX DI oder SI genutzt werden.
– Basisindirekte Adressierung:
Die EA errechnet sich aus der Summe eines Basisregisters, eines Indexregisters und einer Verschiebung.

2. E/A-Adressierung

– Direkte Adressierung:
Die E/A-Adresse wird dem Befehl entnommen. Der E/A-Adreßraum beträgt 256 Byte.
– Indirekte Adressierung:
Das Register DX enthält die E/A-Adresse. Der E/A-Adreßraum ist hier 64 KByte groß.

2.2.4. Befehlssatz

Der Befehlssatz des 8086 kann in folgenden 6 Gruppen eingeteilt werden.

Transportbefehle
Die Gruppe der Transportbefehle realisiert die Übertragung von Bytes oder Wörtern. Es gibt spezifische Befehle zur Behandlung des Stacks (POP, PUSH). Transportbefehle führen keine Speicher-Speicher-Übertragungen durch. Ebenso gibt es keine Bitmanipulationsbefehle. Einen besonderen Befehl dieser Gruppe stellt XLAT dar. Er führt einen tabellenorientierten Transport aus. Das Register AL wird dabei als Index in eine durch BX adressierte, max. 256 Byte lange Tabelle genutzt. Das so ausgewählte Byte wird abschließend in AL abgelegt.

E/A-Befehle
Es können 8- oder 16-Bit-Operanden eingelesen oder ausgegeben werden.

Arithmetisch-logische Befehle

Alle vier Grundrechenarten werden für 8- und 16-Bit-Integer- und ungepackte Dezimalzahlen ausgeführt. Alle logischen Operationen verarbeiten Bytes oder 16-Bit-Wörter. Es stehen die Operationen Negation, UND, ODER, exklusiv ODER, arithmetisch/logisch Verschieben und die Rotation zur Verfügung.

Kettenoperationen

Mit dieser Befehlsgruppe können Zeichenketten effektiv bearbeitet werden. Sie umfaßt Befehle

- zum Transport
- zum Vergleich
- zur Substraktion und
- zum Transport vom oder zum Akkumulator von Zeichenketten.

Steuerungsübergabe

Man kann vier Klassen von Steuerungsübergaben festlegen:

1. CALLs, JMPs, RETs
2. bedingte Übergaben
3. Wiederholungssteuerung
4. Unterbrechungen.

Die erste Gruppe erlaubt unbedingte Programmverzweigungen, das aktuelle Kodesegment kann dabei verlassen werden.

Die zweite Gruppe realisiert Programmverzweigungen in Auswertung vorangegangener arithmetisch-logischer Operationen. Das Verzweigungsziel muß im Bereich von −128 bis +127 Byte von der Absprungstelle liegen. Das aktuelle Kodesegment kann nicht verändert werden.

Die dritte Gruppe erlaubt die Wiederholung von Programmteilen. Die Anzahl der Wiederholungen ist vorher in CX abzulegen. Der mögliche Sprungbereich entspricht dem der Gruppe 2.

Die vierte Gruppe entspricht den schon beschriebenen Softwareinterrupts. Die Unterbrechungen aktualisieren das Kodesegment immer.

Prozessorsteuerung

Die Befehle zur Prozessorsteuerung umfassen folgende Anweisungen bzw. Anweisungsgruppen:

- Anweisungen zur Flagbearbeitung. Es kann das CF-, DF- und OF-Flag gesetzt oder rückgesetzt werden.
- Die Haltanweisung bringt den Prozessor in den Haltzustand, ein Verlassen dieses Zustandes ist nur durch Reset oder eine Unterbrechung möglich.
- Die Ausführung der Waitanweisung erzwingt vom Prozessor so lange Waitzyklen, bis das Signal am TEST-Pin aktiv ist. Damit kann die Arbeit des 8086 mit Coprozessoren synchronisiert werden.
- Die Escapeanweisung erzeugt einen Mechanismus, der es anderen Prozessoren erlaubt, ihre Befehle dem 8086-Befehlsstrom zu entnehmen und die 8086-Adreßmodi zu nutzen. Der 8086 führt diese Anweisungen nicht aus, organisiert aber die Bereitstellung der Operanden.
- Buslock verhindert die Abgabe des Systembusses für den folgenden Befehl.

2.3. Coprozessoren

Der Einsatz von Coprozessoren dient der effektiven Lösung von speziellen Problemen und damit der Erhöhung der Leistungsfähigkeit. In den folgenden Abschnitten sollen der Arithmetik- und der E/A-Prozessor als zwei typische Vertreter kurz vorgestellt werden.

2.3.1. Arithmetikprozessor 8087

Der Arithmetikprozessor übernimmt im MCS-86-System alle Aufgaben der numerischen Verarbeitung von Festkomma-, Gleitkomma- und BCD-Zahlen bis zu einer Breite von 80 Bit. Gemäß der in 2.1. dargelegten Einteilung stellt der Arithmetikprozessor einen Coprozessor dar. In Bild 11 ist zu erkennen, daß der 8087 zum 8086 parallel geschaltet ist /4/. Beide Prozessoren nutzen die Systembusinterfacebausteine gemeinsam. Der Zugriff auf den gemeinsamen lokalen Bus wird durch die Request-/Grant-Leitungen (/RQ/ /GT) zwischen beiden MPs gesteuert. Aktiviert wird der Arithmetikprozessor durch die Ausführung eines Escape-Befehls. Dazu hört der 8087 den Befehlsstrom des 8086 nach solch einer Anweisung ab. Findet er eine, so ist er mittels der Analyse der Befehlsschlangenstatussignale (/QS0, /QS1) in der Lage, zu erkennen, wann der Escapebefehl abgearbeitet wird. Die Synchronisation beider Prozessoren geschieht über die TEST-Leitung des 8086, so daß er erkennen kann, wann der 8087 eine Operation beendet hat. Zur Signalisierung von Ausnahmezuständen (z. B. Division durch Null) kann der 8087 einen maskierbaren Interrupt auslösen.

Der Befehlssatz umfaßt die folgenden sechs Gruppen:

Transportbefehle

Sie dienen zum Laden und Speichern der Register des 8087. Da intern immer mit einer Breite von 80 Bit gearbeitet wird, findet hier auch sofort eine Konvertierung in die entsprechenden Formate statt.

Arithmetische Befehle

In dieser Gruppe befinden sich die Befehle zur Ausführung der Grundrechenarten, der Bildung der Quadratwurzel, des Vorzeichenwechsels, der Absolutwertbildung und weiterer einfacher mathematischer Funktionen.

Logische Operationen

Die Logikbefehle beinhalten den Vergleich von Variablen und die Trennung von Exponent und Mantisse bei Gleitkommazahlen.

Transzendentale Funktionen

Als transzendentale Funktionen stehen der Tangens, der Arcustangens und verschiedene Logarithmenfunktionen zur Verfügung.

Konstantenbereitstellung

Ausgewählte Konstanten, wie 0, 1, pi, lg2, ln2, ld10, lde, sind im 8087 direkt abrufbar.

Prozessorsteuerung

Diese Befehlsgruppe ermöglicht die Steuerung des 8087. Folgende Funktionen sind ausführbar:

- Rundung

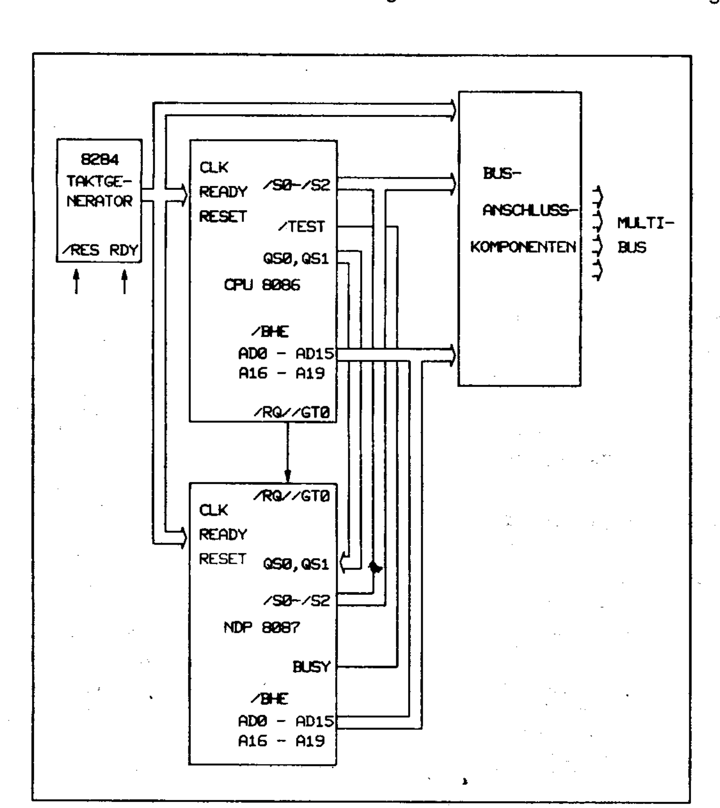


Bild 11 MCS-86 System mit Arithmetikprozessor

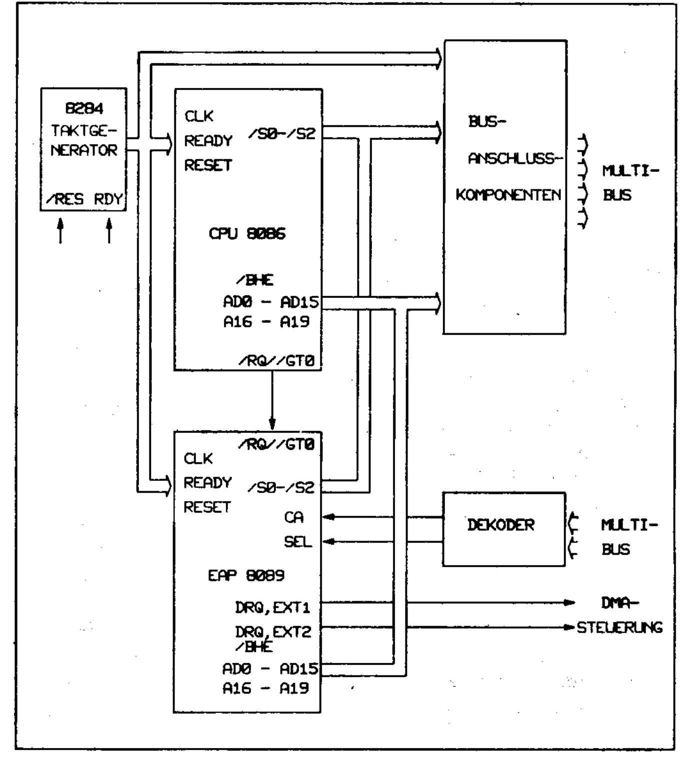


Bild 12 EAP 8089 im lokalen Arbeitsmode

– Fehlerbehandlung
– Abspeichern des gesamten Registersatzes
– Laden des gesamten Registersatzes.

Abschließend sind die vom 8087 verarbeitbaren Datentypen aufgelistet:

Integer:	
Wort	16 Bit
Short	32 Bit
Long	64 Bit
BCD gepackt:	
18 Stellen	
Real:	
Short	32 Bit
Long	64 Bit
Temporary	80 Bit.

Intern werden alle Operationen mit 80 Bit Breite ausgeführt. Speicheroperanden können nur eine Breite von 8 Byte haben.

2.3.2. E/A-Prozessor (EAP) 8089

Trotz integrierter Peripheriebausteine, wie PIO, SIO und CTC, stellen die Steuerung dieser Bausteine und die Datenübertragung eine hohe Belastung für die CPU dar. Eine Entlastung der CPU brachte die Entwicklung von DMA-Controllern. Diese werden aber immer noch in Zeitteilung mit der CPU betrieben und müssen von ihr überwacht werden. Eine völlig neue Lösung stellt die Entwicklung von E/A-Prozessoren dar. Der zum MCS-86 gehörende unabhängige Prozessor 8089 entlastet die CPU von zeitaufwendigen Transportaufgaben.

2.3.2.1. Arbeitsmodi

Der EAP 8089 kann in zwei Arbeitsmodi betrieben werden:
1. lokale Betriebsart
2. unabhängige Betriebsart.

In der im Bild 12 dargestellten lokalen Betriebsart wird der EAP ähnlich dem 8087 zur CPU parallel geschaltet. Durch die Zeitteilung am gemeinsamen lokalen Bus kann hier natürlich die volle Systemleistung nicht erreicht werden. Der Buszugriff wird wieder über die /RQ/GT-Leitung gesteuert. Der EAP erfüllt in solchen Konfigurationen vor allem zeitaufwendige Transportoperationen mit DMA-Geschwindigkeit.

Bild 13 stellt den EAP in der unabhängigen Betriebsart dar /5/. Er verfügt in diesem Arbeitsmode über einen eigenen lokalen und einen Anschluß an den Multibus. Der Busarbiter und der Busverwalter lassen sich durch geeignete Beschaltung so programmieren, daß alle Speicheroperationen über den Multimasterbus und alle E/A-Operationen über den lokalen Bus ablaufen. Der EAP selbst kann sein Programm aus dem Systemspeicher oder seinem lokalen E/A-Bereich beziehen. Damit wird eine volle Parallelarbeit zur CPU möglich.

Für beide Arbeitsmodi gilt, daß der EAP immer sein eigenes Programm abarbeitet. Er empfängt seine Befehle also nicht aus dem Befehlsstrom des 8086.

2.3.2.2. Architektur und Leistungsmerkmale

Der 8089 kann, wie auch der 8086, einen Speicherraum von 1 MByte und einen E/A-Adreßraum von 64 MByte direkt ansprechen. Die Busbreite (8 oder 16 Bit) ist für beide Busse, den Systembus und den lokalen Bus, getrennt programmierbar. Es sind auch Mischvarianten möglich, so zum Beispiel ein 16-Bit-Systembus und ein 8 Bit breiter lokaler Bus. Die Konvertierung beider ineinander führt der EAP automatisch aus. Die zwei unabhängigen DMA-Kanäle werden durch je ein Signal aktiviert. Im Ruhezustand besteht also keine Busbelastung. Der Registersatz dieses Prozessors ist in Bild 14 dargestellt /5/. Er ist für jeden Kanal einmal vorhanden. Die Register GA, GB und GC sind Adreßregister und besitzen eine Breite von 21 Bit. Im Normalbetrieb sind sie universell verwendbar, während DMA-Operationen beinhalten sie die Ziel- und Quelladresse sowie die Adresse für eine mögliche Übersetzungstabelle. Mittels dieser Übersetzungstabelle kann während des DMA-Betriebs eine Kodetransformation vorgenommen werden. Das Bit 20 dient als Zeiger, ob sich die adressierte Speicherzelle am lokalen Bus oder am Systembus befindet.

Das TP-Register entspricht dem Befehlszähler.

Ein zweiter Block enthält ein Index-, ein Zähl-, ein Masken- und ein Kanalsteuerregister. Außer dem Kanalsteuerregister können alle diese 16-Bit-Register bei Nicht-DMA-Betrieb universell eingesetzt werden. Das PP-Register enthält die Adresse des Parameter-

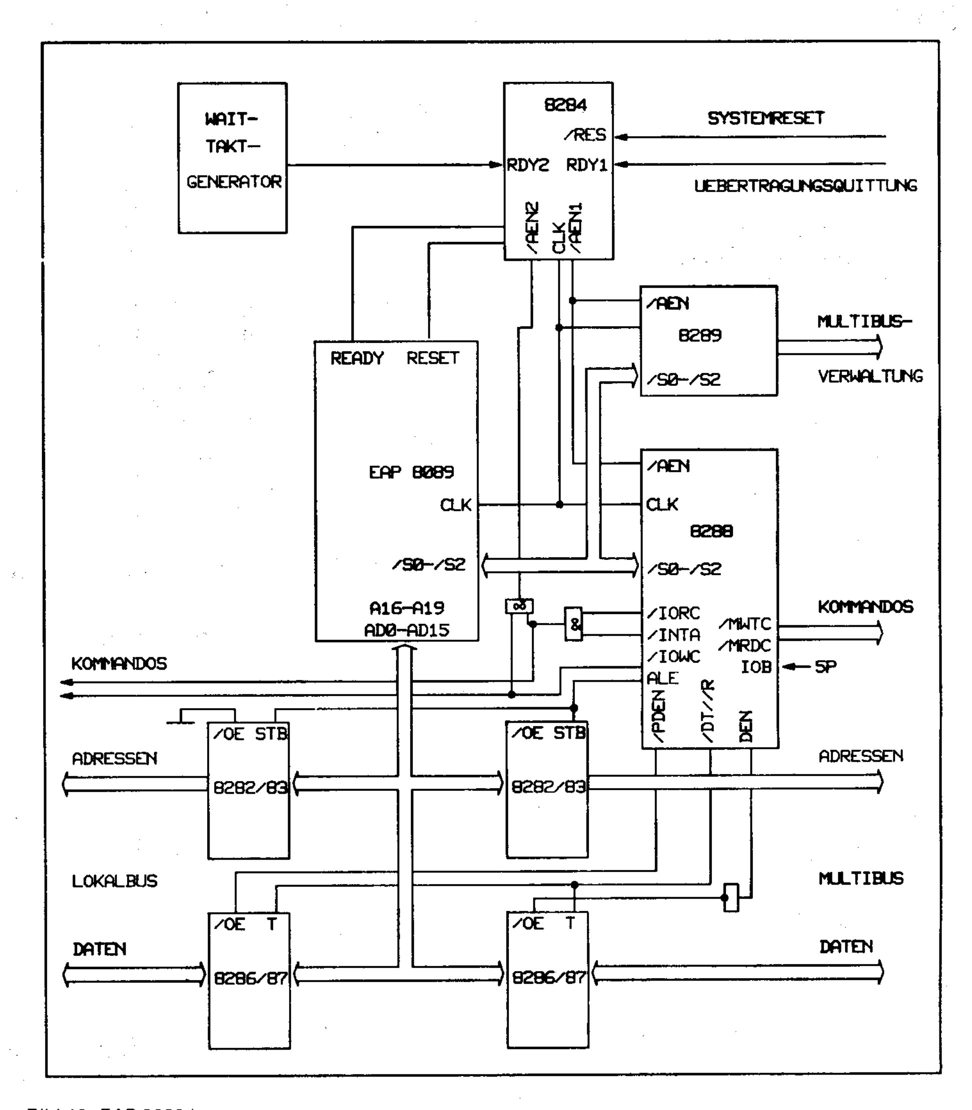


Bild 13 EAP 8089 in der unabhängigen Betriebsart

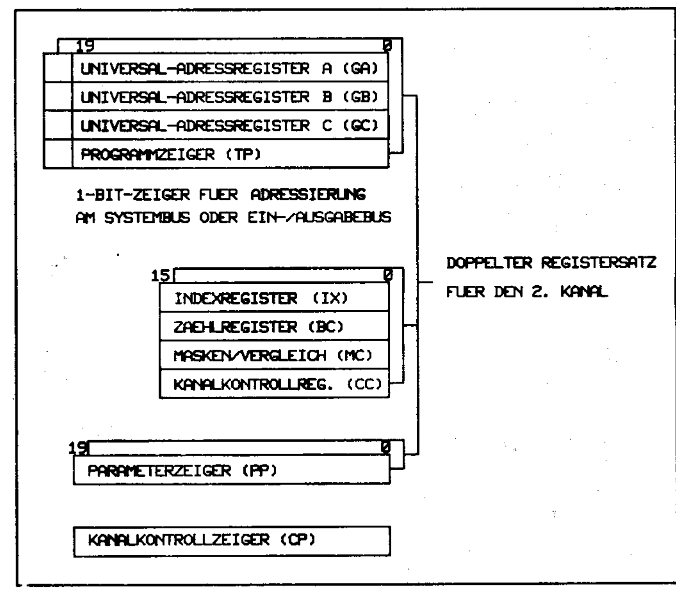


Bild 14 Registersatz des EAP 8089

blocks, welcher zur Kommunikation mit der CPU genutzt wird. Die Adresse des Steuerblocks wird in dem Register CP abgelegt. Es ist nur einfach vorhanden.
Die Initialisierung des EAP geschieht in mehreren Ebenen und wird von der CPU gestartet. Dabei liest der EAP durch Adressen verkoppelte Steuerblöcke ein und findet so die Startadresse der Kanalprogramme. Der EAP besitzt einen eigenen Befehlssatz und benötigt zur Übersetzung einen eigenen Assembler, welcher nicht im 8086-Assembler enthalten ist.

Literatur

/1/ Matasatoshi, S.: 16-Bit-Chip genügt Mikro- und Minicomputeranforderungen. Elektronik 1979, Heft 8, S. 83–87
/2/ Roth, M.: Mikroprozessoren. Wissenschaftliche Zeitschrift der TH Ilmenau, Ilmenau 1979
/3/ 8086 User's manual. Intel Corp. 1979
/4/ Dorn, L.: Mathematik-Coprozessor: hundertmal schneller als Software. Elektronik 1981, Heft 24, S. 57–61
/5/ Dorn, L.: Ein neuer Mikroprozessortyp: Ein-/Ausgabeprozessor. Elektronik 1980, Heft 3, S. 75–80

✉ KONTAKT ☏

Technische Universität Dresden, Sektion Informationstechnik, Bereich Kommunikations- und Computertechnik, Mommsenstr. 13, Dresden, 8027; Tel. 4 63 22 30

RAM-Disk für K-1520-Systeme

Wolfram Kammer, Wolfgang Spindler
VEB Elektronische Bauelemente Teltow

RAM-Disks sind nützliche Einrichtungen, die wohl besonders von denjenigen sehr geschätzt werden, die an Rechnern mit Kassettenbandspeichern ihre Geduld üben müssen. Aber auch im Vergleich zur Diskette kann die RAM-Disk erstaunliche Geschwindigkeit bieten. Wer schon einmal in längeren Assemblerquellen mit einem Textverarbeitungsprogramm wie TP herumgesucht hat, weiß ein Lied davon zu singen. Hier wird nun eine konkrete Schaltung einschließlich Layout vorgestellt, die außer einer RAM-Disk von 256 KByte auch noch einen kompletten Hauptspeicher sowie eine MEMDI-Erzeugung enthält. Bild 1 zeigt die Schaltung. Die eigentliche RAM-Disk wird über IN- und OUT-Befehle bedient und tangiert den Hauptspeicher nicht.

Funktion der RAM-Disk

Zum Adressieren von 256 KByte werden 18 Adreßbits benötigt. Die niederwertigsten 8 Bit stellt ein vom Programm mittels OUT-Befehl ladbarer Adreßzähler (2 × 74LS193) bereit. Die nächsthöheren 8 Bit müssen vom Programm in ein Oktallatch (DS8282) geladen werden. Die restlichen 2 Bit stecken in der Peripherieadresse, unter der das Bedienprogramm anschließend die RAM-Disk liest oder beschreibt. Nach jedem Zugriff inkrementiert die Zugriffslogik der RAM-Disk den o.g. Adreßzähler. Damit sind INIR- und OTIR-Befehle für das Umladen der Daten bestens geeignet. Die RAM-Disk belegt insgesamt 8 E/A-Adressen nach folgendem Schema:
Grundadresse plus
0 = Lesen/Schreiben Bank 1
1 = Lesen/Schreiben Bank 2
2 = Lesen/Schreiben Bank 3
3 = Lesen/Schreiben Bank 4
4 = nicht benutzen
5 = nicht benutzen
6 = mittlere 8 Adreßbit laden
7 = niedrigste 8 Adreßbit in den Zähler laden.
Die Grundadresse kann man in gewissen Grenzen frei wählen, indem man das Wickelfeld D entsprechend verdrahtet.

```
RAMDSK  MACRO-80 3.4     01-Dec-80        PAGE     1

 0000'                           CSEG
                                 .Z80
                                 TITLE  RAMDSK
                          ;
                          ; BEDIENPROGRAMM FUER RAM-DISK
                          ;.
                          ; VORGABEBEREICHE IM BIOS
 0000'                    QTRAC: DS 2        ; SPURNUMMER
 0002'                    QSECT: DS 2        ; SEKTORNUMMER
 0004'                    QDMA:  DS 2        ; ADR.PUFFERBEREICH
 0006'                    DIRBF: DS 128      ; BDOS-PUFFER
                          ;
                          ;GRUNDADRESSE DER RAM-DISK
 00E0                     GADDR  EQU 0E0H
                          ;
                          ; LESEN EINES SEKTORS ZU 128 BYTE
 0086'    CD 0094'        LESE:  CALL ADRE
 0089'    ED B2                  INIR
 008B'    AF                     XOR  A
 008C'    C9                     RET
                          ;
                          ; SCHREIBEN EINES SEKTORS ZU 128 BYTE
 008D'    CD 0094'        SCHRB: CALL ADRE
 0090'    ED B3                  OTIR
 0092'    AF                     XOR  A
 0093'    C9                     RET
                          ;
                          ; ADRESSRECHNUNG FUER LESEN+SCHREIBEN
 0094'    2A 0000'        ADRE:  LD   HL,(QTRAC)
 0097'    AF                     XOR  A
 0098'    29                     ADD  HL,HL
 0099'    29                     ADD  HL,HL
 009A'    29                     ADD  HL,HL
 009B'    29                     ADD  HL,HL
 009C'    ED 5B 0002'            LD   DE,(QSECT)
 00A0'    19                     ADD  HL,DE
 00A1'    CB 1C                  RR   H
 00A3'    CB 1D                  RR   L
 00A5'    CB 1F                  RR   A
 00A7'    D3 E7                  OUT  (GADDR+7),A
 00A9'    7D                     LD   A,L
 00AA'    D3 E6                  OUT  (GADDR+6),A
 00AC'    7C                     LD   A,H
 00AD'    E6 03                  AND  3
 00AF'    F6 E0                  OR   GADDR
 00B1'    4F                     LD   C,A
 00B2'    06 80                  LD   B,128
 00B4'    2A 0004'               LD   HL,(QDMA)
 00B7'    C9                     RET
                          ;
                          ; INITIALISIEREN BEI KALTSTART
 00B8'    AF              INIT:  XOR  A
 00B9'    4F                     LD   C,A
 00BA'    D3 E6                  OUT  (GADDR+6),A
 00BC'    AF              INI1:  XOR  A
 00BD'    47                     LD   B,A
```

Bild 2 Bedienprogramm für die RAM-Disk

```
RAMDSK  MACRO-80 3.4     01-Dec-80        PAGE     1-1

 00BE'    D3 E7                  OUT  (GADDR+7),A
 00C0'    3E E5                  LD   A,0E5H
 00C2'    D3 E0           INI2:  OUT  (GADDR),A
 00C4'    10 FC                  DJNZ INI2
 00C6'    0C                     INC  C
 00C7'    79                     LD   A,C
 00C8'    D3 E6                  OUT  (GADDR+6),A
 00CA'    FE 09                  CP   9
 00CC'    38 EE                  JR   C,INI1
 00CE'    C9                     RET
                          ;
                          ; DISK-PARAMETER-HEADER
 00CF'    0000            DPHE:  DW   0     ;NO TRANSLATION TABLE
 00D1'    0000                   DW   0     ; 6 BYTE ARB.ZELLEN
 00D3'    0000                   DW   0     ; FUER BDOS
 00D5'    0000                   DW   0
 00D7'    0006'                  DW   DIRBF ;128 BYTE PUFFER
 00D9'    00DF'                  DW   DPB   ;DISK-PARAMETER-BLOCK
 00DB'    00EE'                  DW   CSV   ;CHECKSUMMENBEREICH
 00DD'    00EF'                  DW   ALV   ;BLOCKBELEGUNGSPLAN
                          ;
                          ; DISK-PARAMETER-BLOCK
 00DF'    0010            DPB:   DW   16    ;SEKTOREN PRO SPUR
 00E1'    03                     DB   3     ;BLOCKSHIFTFAKTOR
 00E2'    07                     DB   7     ;BLOCKMASK
 00E3'    00                     DB   0     ;EXTENTMASK
 00E4'    00FF                   DW   255   ;BLOCKANZAHL-1=256 K
 00E6'    003F                   DW   3FH   ;64 DIR-EINTRAGUNGEN
 00E8'    00C0                   DW   0C0H  ;DIRECTORY-BLOECKE
 00EA'    0000                   DW   0     ;KEIN DISK-CHECK
 00EC'    0000                   DW   0     ;KEINE SYSTEMSPUREN
                          ;
                          ; CHECKSUMMENBEREICH FUER DEN TEST
                          ; VON BDOS AUF DISKETTENWECHSEL
 00EE'    00              CSV:   NOP        ;KEIN TEST:FESTPLATTE
                          ;
                          ; BELEGUNGSPLAN DER DISK
                          ; (ALLOCATION-VECTOR)
                          ; JEDES BIT ENTSPRICHT EINEM BLOCK
                          ; (32*8=256 BLOECKE ZU JE 1 K BYTE)
 00EF'                    ALV:   DS   32
                          ;
                          END
```

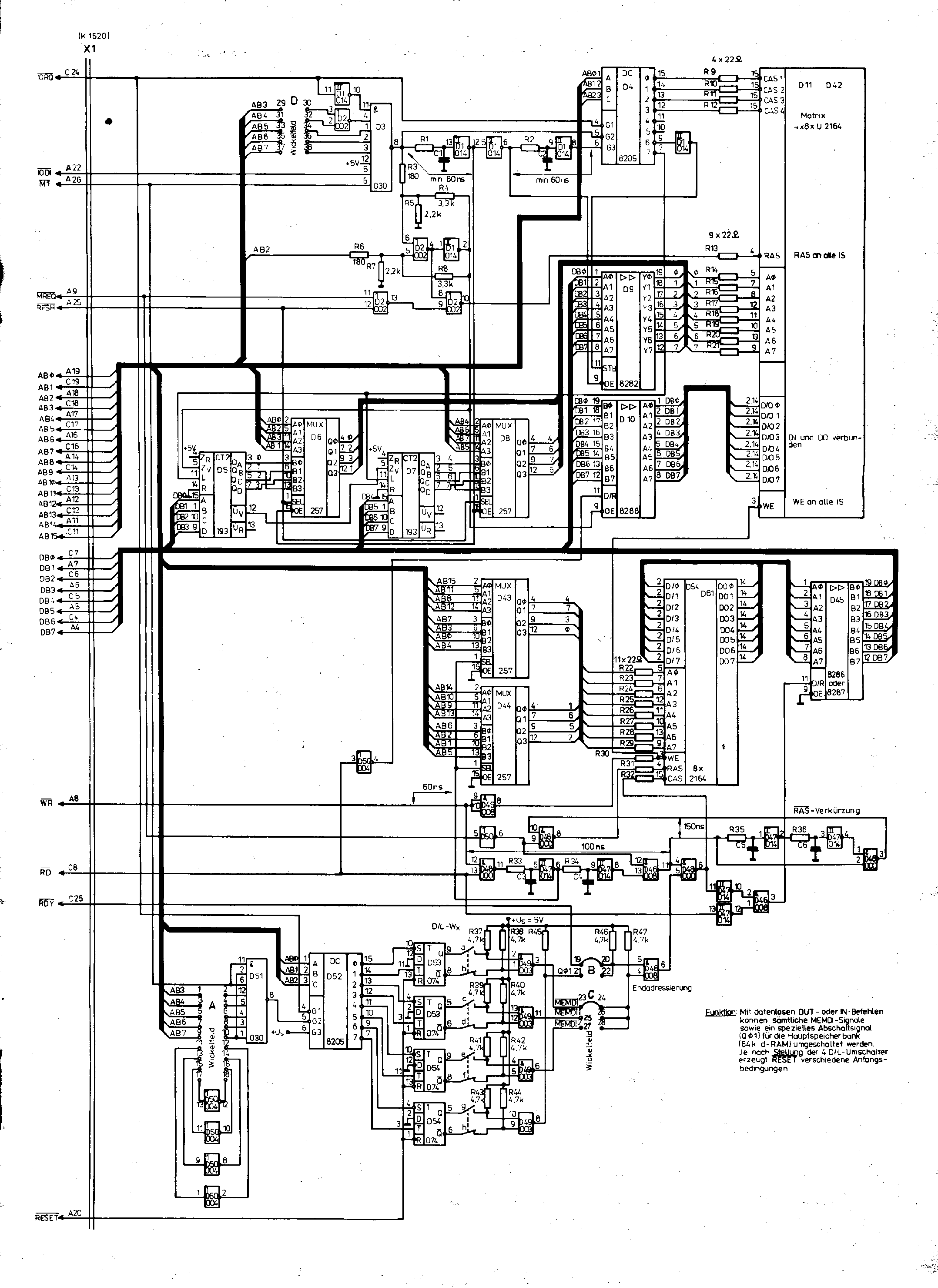


Bild 1 Schaltplan der RAM-Disk

Diplomingenieur Wolfram Karner (32) studierte von 1978 bis 1982 an der Ingenieurschule Dresden, Fachrichtung medizinische Gerätetechnik. Seit 1982 ist er im Bereich Bauelementeentwicklung des VEB Elektronische Bauelemente Teltow tätig.

Diplomphysiker Wolfgang Spindler (36) studierte von 1973 bis 1978 an der Technischen Universität Dresden. Im Anschluß daran Tätigkeit im VEB Gleichrichterwerk Stahnsdorf in der Meßtechnikentwicklung. Seit 1982 ist er im Bereich Elektronik Mikrorechentechnik des VEB Elektronische Bauelemente Teltow tätig.

Block (KByte)	Blockshiftfaktor	Blockmask
1	3	07H
2	4	0FH
4	5	1FH
8	6	3FH
16	7	7FH

Der Wert der "Extentmask" ergibt sich aus der Blockgröße und der Speicherkapazität der Disk:

Block (KByte)	Extentmask bei Disk<256K	Disk>256K
1	0	geht nicht
2	1	0
4	3	1
8	7	3
16	15	7

Bild 3 Verwendbare Blockgrößen

Funktion des Hauptspeichers

Der Hauptspeicher umfaßt volle 64 KByte. Damit außer ihm auch noch andere Baugruppen (Urlader, Bildschirm usw.) im Speichervolumen betrieben werden können, besitzt er eine Einrichtung zum Entadressieren.
Dazu besitzt die Baugruppe einen Eingang für das K1520-Bussignal READY. Aktiviert bei einem Speicherzugriff eine andere Baugruppe diese Leitung, so tritt der Hauptspeicher in dem Bereich, den die andere Baugruppe belegt, in den Hintergrund, indem der Lese- oder Schreibzugriff in einen Refreshzyklus umgewandelt wird. Dafür stehen bei Lesezyklen ca. 100 ns und bei Schreibzyklen etwas mehr als 1 Systemtakt zur Verfügung. Schaltet man mittels MEMDI-Signalen die anderen Baugruppen ab, so kommt automatisch der Hauptspeicher wieder hervor. Dieses Verfahren erleichtert die Speicherverwaltung ganz erheblich.
Um den Hauptspeicher auch bei 4 MHz Systemtakt ohne die lästigen WAIT-Zyklen betreiben zu können, schaltet die Zugriffslogik das RAS-Signal ca. 150 ns nach Aktivwerden des CAS-Signales ab. Damit wird den Speicherschaltkreisen genügend Zeit zum Rückschreiben der Information in die Matrix verschafft. Das ist besonders wichtig für die sehr kurze Zugriffslücke zwischen M1- und Refreshzugriff beim Befehlsholezyklus des U880.

Funktion der MEMDI-Erzeugung

Dieser Schaltungsteil besteht aus 4 Flipflops, die über 8 OUT-Adressen einzeln gesetzt und gelöscht werden können. Die von den OUT-Befehlen ausgegebenen Daten sind dabei belanglos. Durch Systemreset werden alle vier Flipflops gelöscht. Welche MEMDI-Signale dabei gesetzt/gelöscht werden, kann man durch DIL-Schalter oder Wickelbrücken frei wählen. Leider war auf der Leiterplatte nicht mehr genügend Platz, so daß der Koppelbussteckverbinder entfallen mußte. Deshalb müssen die MEMDI-Signale am Wickelfeld C abgegriffen werden.

Zum Aufbau

Auf der Leiterplatte sind einige Stützkondensatoren (33 nF Epsilan sowie Elkos 47 μF/6,3 V) vorgesehen, die im Stromlaufplan nicht extra angegeben wurden. Sie sind mit Cs bezeichnet.
Alle Logikschaltkreise (außer Bustreiber und

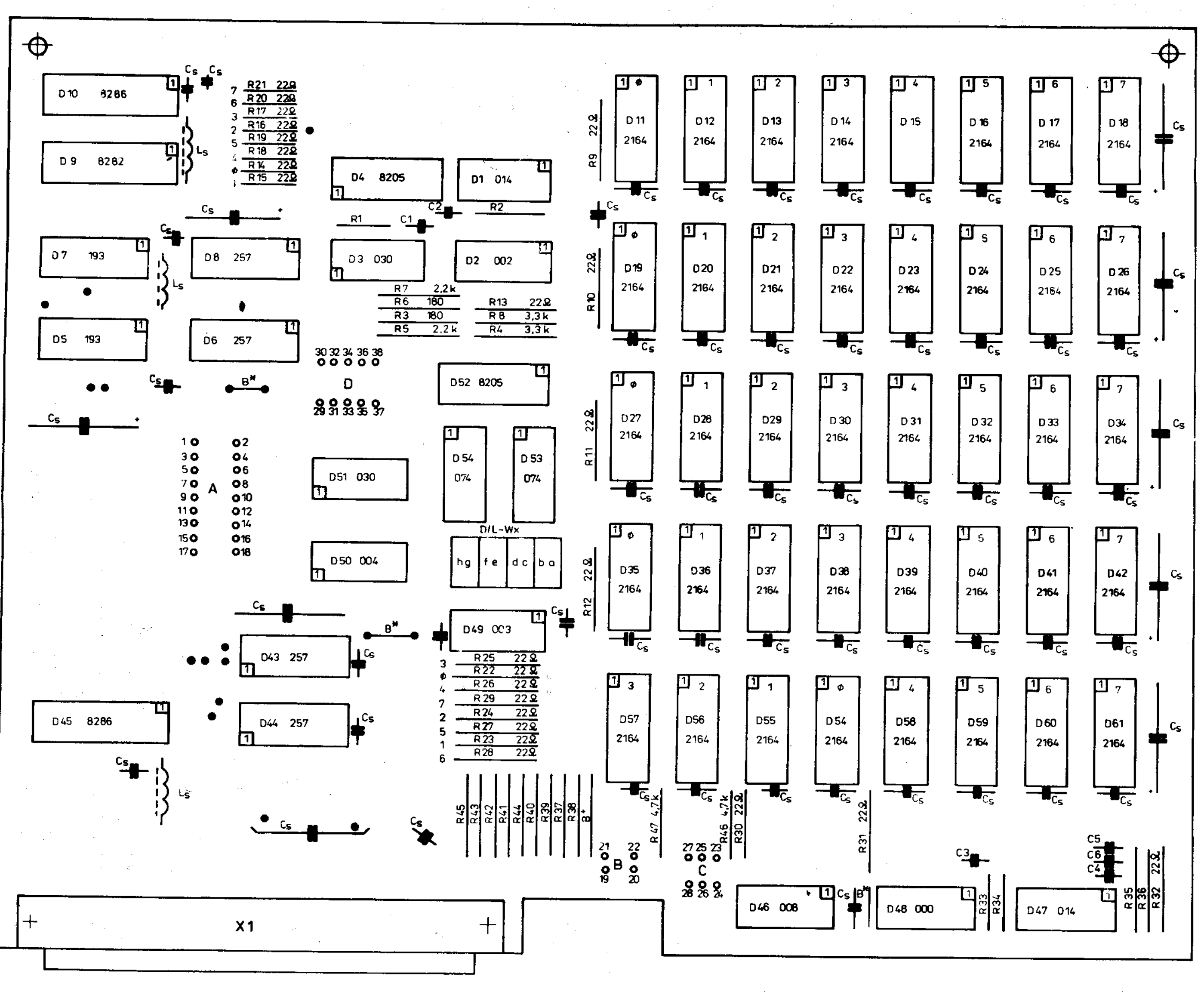


Bild 4 Bestückungsplan

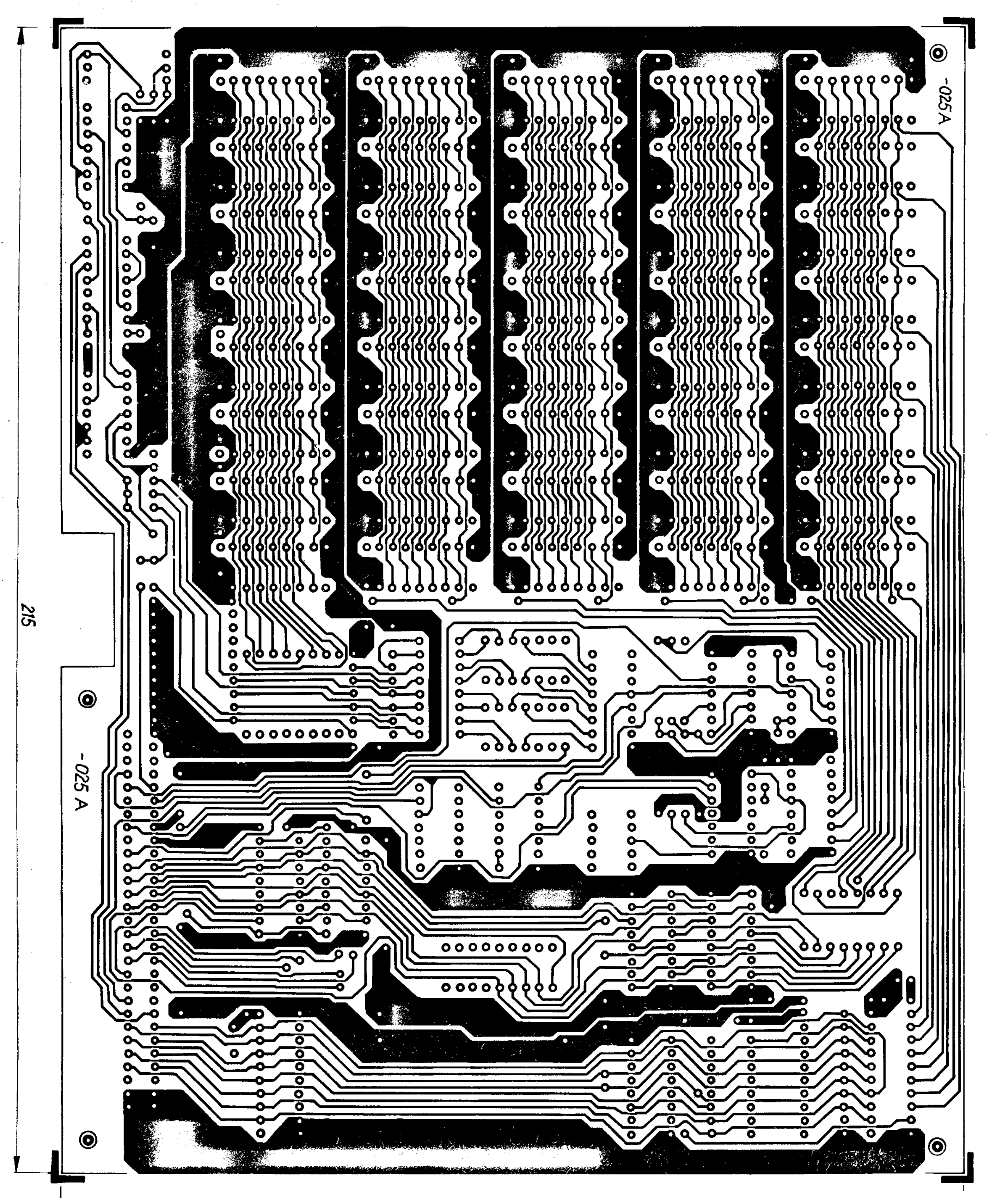
Dekoder) sind Low-Power-Schottky-, also 74LSxxx-Typen. Im Stromlaufplan wird nur die Typnummer (also 193 entspricht 74LS193) angegeben.
Als Speicherschaltkreise können alle U2164-Typen eingesetzt werden.
Die Verzögerungsglieder in den RAS/CAS-Schaltungen wurden zu 390 pF und 100 ... 180 Ω gewählt.

Einbindung der RAM-Disk in das BIOS

CP/M und ähnliche Plattenbetriebssysteme bestehen im wesentlichen aus 3 Teilen:
1. dem BIOS (Basis-Ein/Ausgabesystem)
2. dem BDOS (eigentliches Plattenbetriebssystem)
3. dem CCP (Kommandoprogramm).

Zur Einbindung der RAM-Disk muß das in Bild 2 gezeigte Bedienprogramm in das BIOS eingefügt werden. Bei den meisten Systemen findet man im BIOS zwei oder drei Tabellen, in denen die Adressen der Disk-Parameter-Header (DPH) sowie die Adressen der zugehörigen Lese- und Schreibroutinen stehen. Dort muß man die eingefügten Programmteile eintragen. Wer mehr als 256

Bild 5 Layout Lötseite der RAM-Disk



KByte RAM-Disk einbauen will, muß allerdings den Disk-Parameter-Block (DPB) ändern. Da das BDOS jedes Laufwerk in maximal 256 Blöcke aufteilt, ist bei größeren Speichern die Blockgröße entsprechend Bild 3 im DPB zu vereinbaren. Wenn die RAM-Disk richtig ins BIOS eingebaut ist, so ergibt die Überprüfung mit STAT DSK:

folgende Antwort:

```
   E: Drive Characteristics
2048: 128 Byte Record Capacity
 256: Kilobyte Drive Capacity
  64: 32 Byte Directory Entries
   0: Checked Directory Entries
 128: Records/ Extent
   8: Records/ Block
  16: Sectors/ Track
   0: Reserved Tracks
```

✉ KONTAKT ☏

VEB Elektronische Bauelemente Teltow, Abt. EVP, Ernst-Thälmann-Str. 10–15, Teltow, 1530; Tel. 45 – 34 59

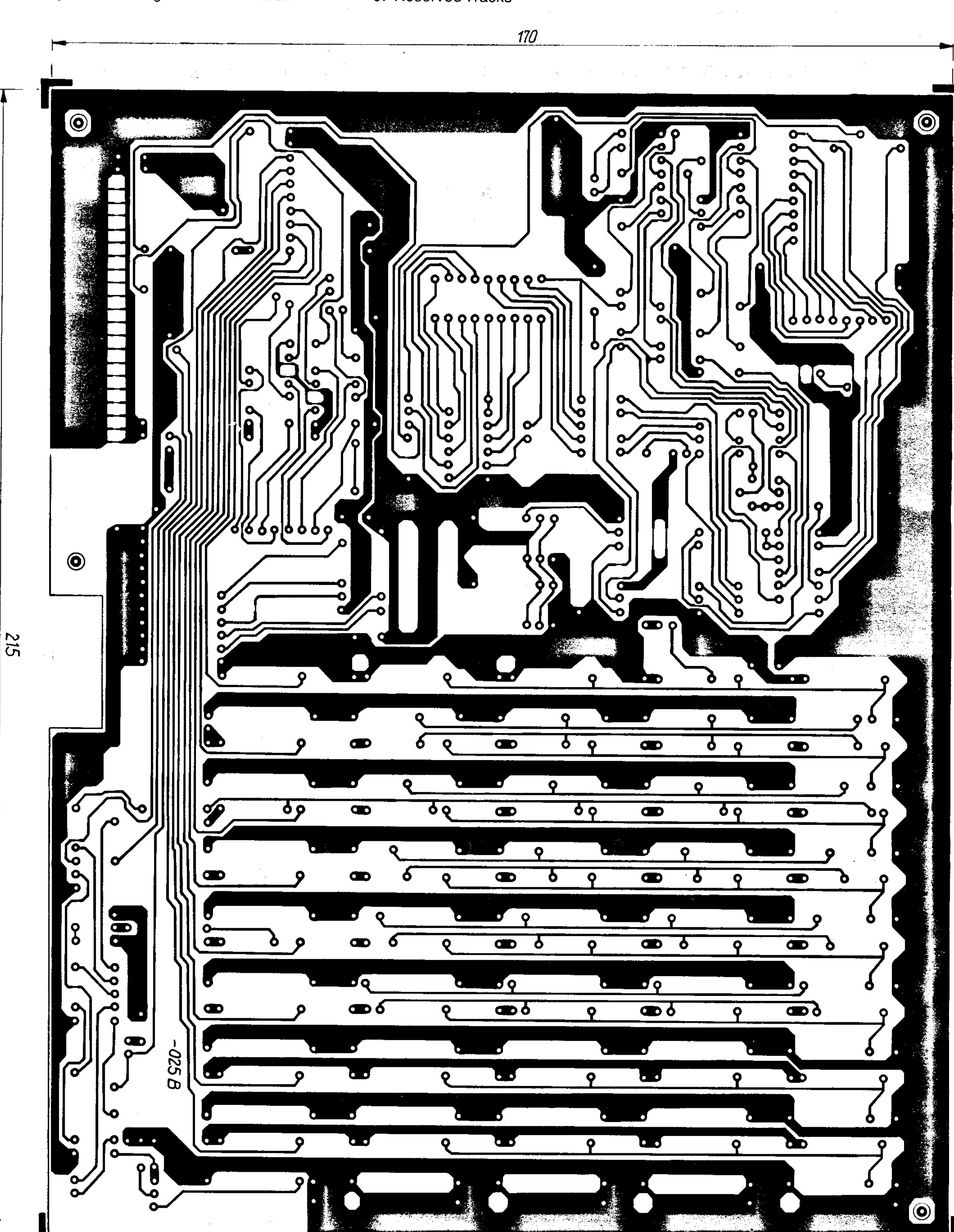


Bild 6 Layout Bestückungsseite der RAM-Disk

PASCAL (Teil 3)

Dr. Claus Kofer
Informatikzentrum des Hochschulwesens an der Technischen Universität Dresden

Der Deklarationsteil der Funktion ist wieder leer. In der If-Anweisung des Anweisungsteils wird dem Funktionsnamen ein Wert zugewiesen. Auf diesen Wert kann durch Anruf der Funktion Max2 Bezug genommen werden. Zum Beispiel in folgender Form:

```
Z:=Max2(x,y)
Z:=Max2(1.0,2.5*y)
IF Z>Max2 (1.5*x,y) THEN ...
REPEAT ... UNTIL Max2(x,y)<1.0
```

Es wird jetzt eine Prozedur Swap gezeigt, die die Werte ihrer Parameter vertauscht.

```
PROCEDURE Swap (VAR A,B:CHAR);
VAR H:CHAR
BEGIN
  H:=A;  A:=B;  B:=H
END;
```

Im Prozedurkopf werden wieder Name und formale Parameter angegeben. Es sind beides Referenzparameter. Die verwendete Schreibweise ist eine Abkürzung für PROCEDURE Swap (VAR A:CHAR; VAR B:CHAR). Da das Vertauschen ohne eine Hilfsvariable nicht möglich ist, wird diese im Deklarationsteil der Prozedur Swap deklariert. Der Anweisungsteil besteht aus einer Liste von drei Zuweisungen, die das Vertauschen bewerkstelligen.
Die aktuellen Parameter der Prozedur Swap dürfen nur Variablen vom Typ CHAR sein:

```
Swap (C1,C2)
```

Aber nicht

```
Swap ('A',C1)
```

Für die Verwendung von Funktionsparametern wird nun ein Beispiel angegeben:

```
FUNCTION Diff(XO,DX:REAL;
      Fkt(x:REAL):REAL):REAL;
BEGIN
  Diff:=(Fkt(XO+DX)-Fkt(XO))/DX
END;
```

Der Funktionskopf von Diff enthält neben der bereits bekannten Deklaration der Werteparameter XO und DX einen formalen Funktionsparameter Fkt. Laut Deklaration ist Fkt eine Funktion mit einem Parameter vom Typ REAL und liefert selbst einen Funktionswert vom Typ REAL. Im Anweisungsteil wird der Wert der Funktion Diff als Differenzenquotient der Funktion Fkt für das Argument XO berechnet.

Als aktueller Parameter darf für Fkt jede beliebige Funktion verwendet werden, die einen Parameter vom Typ REAL hat und selbst einen Wert vom Typ REAL liefert. Es werden zwei solcher Funktionen gezeigt:

```
FUNCTION P1(x:REAL):REAL;
BEGIN
  P1:= (1.5*x+2.5)*x+3.5
END;

FUNCTION P2(x:REAL):REAL
BEGIN
  P2:=1/((0.5*x+0.25)*x
END;
```

Unter Verwendung von P1 und P2 sind nun folgende Aufrufe der Funktion Diff möglich

```
A:=Diff(x,0.01,P1);
A:=Diff(x,0.01,P2)
```

Leider gibt es bei den meisten PASCAL-Systemen für Prozedur- und Funktionsparameter die Einschränkung, daß als aktuelle Parameter nicht die sogenannten Standardprozeduren und -funktionen verwendet werden dürfen.
Bei TURBO-PASCAL sind Prozedur- und Funktionsparameter überhaupt nicht implementiert.

5.5.3. Blockstruktur, Gültigkeitsbereiche von Bezeichnern

Nach den Syntaxdiagrammen der Bilder 5.1, 5.2, 5.24 und 5.26 können Prozeduren und Funktionen selbst wieder weitere Prozeduren und Funktionen enthalten. Es entsteht eine Struktur von ineinander verschachtelten Blöcken, von denen jeder aus einem (möglicherweise leeren) Deklarationsteil und einem Anweisungsteil besteht.
Bild 5.31 zeigt eine typische Blockstruktur. Die Prozeduren A und C sind Bestandteil der Deklarationen des Hauptprogramms. Der Deklarationsteil der Prozedur A enthält die Prozedur B.
Es sind beliebige Schachtelungen von Prozeduren und Funktionen denkbar, allerdings begrenzen die PASCAL-Systeme oft deren Tiefe. Im Bild 5.31 ist die größte Schachtelungstiefe drei: Hauptprogramm, Prozeduren A und B.
In jedem Block können Marken, Konstanten, Datentypen, Variablen sowie weitere Prozeduren und Funktionen deklariert werden. Das wirft die Frage nach den Gültigkeitsbereichen der Bezeichner auf. Es gibt dafür zwei einfache Regeln:
1. Ein Bezeichner gilt im Inneren des gesamten Blocks, in dem er deklariert wurde.
2. Bei gleichen Bezeichnern gilt die lokalere Deklaration.
Der Name einer Prozedur bzw. Funktion gehört zu den Bezeichnern des umschließenden Blocks. Die möglicherweise vorhandenen formalen Parameter sind jedoch als Bestandteil des Deklarationsteils der Prozedur bzw. Funktion anzusehen.
Die Gültigkeitsbereiche der Bezeichner des Programmfragments aus Bild 5.31 werden tabellarisch im Bild 5.32 gezeigt. Die Bezeichner T und A des Hauptprogramms gelten in allen Blöcken. K gilt nur in den Blöcken PROG und C. In den Blöcken A und B wird K durch

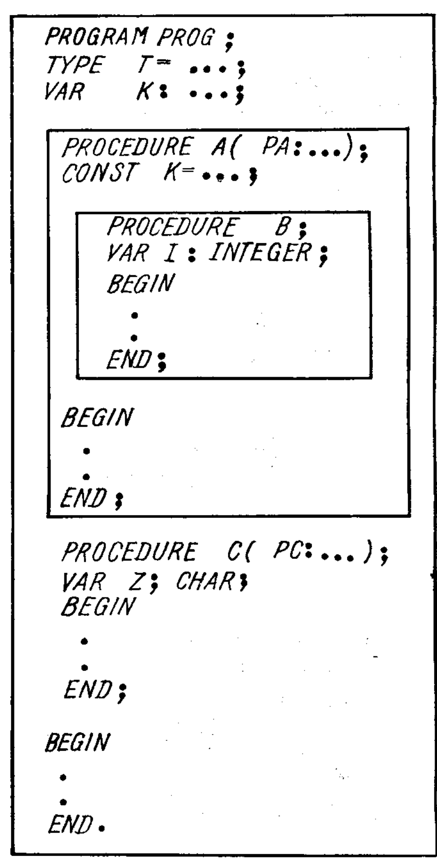


Bild 5.31 Blockstruktur eines Programms

		PROG	A	B	C
PROG	T	(1)	(1)	(1)	(1)
	K	(1)			(1)
	A	(1)	(1)	(1)	(1)
	C	(1)			(1)
A	PA		(1)	(1)	
	K		(1,2)	(1,2)	
	B		(1)	(1)	
B	I			(1)	
C	PC			(1)	
	Z			(1)	

Bild 5.32 Gültigkeitsbereiche der Bezeichner aus Bild 5.31. Die Zeilen korrespondieren mit den Bezeichnern, die Spalten mit den Blöcken. Eine Ziffer zeigt an, aufgrund welcher Regel ein Bezeichner in einem Block gilt.

eine Konstante mit dem gleichen Bezeichner überdeckt.
Der Bezeichner C gilt nur in den Blöcken PROG und C. In den Blöcken A und B ist C nicht bekannt, weil C noch nicht deklariert ist.
Die Bezeichner PA, K und B der Prozedur A gelten in den Blöcken A und B. Außerhalb des Blockes A sind sie nicht bekannt.
Der Bezeichner I der Prozedur B gilt nur im Block B.
Die Bezeichner PC und Z der Prozedur C gelten im Block C. Bild 5.31 zeigt, daß im Block C auch der Bezeichner A bekannt ist, denn seine Deklaration erfolgte im Programmtext schon vor dem Block C.
Für jeden Block können lokale und globale Deklarationen unterschieden werden. Lokale Deklarationen werden durch den Block selbst eingebracht. Globale Deklarationen stammen aus einschließenden Blöcken.
Die global deklarierten Variablen stellen eine weitere Möglichkeit zum Datenaustausch zwischen Prozedur oder Funktion und ihrer Umgebung dar. Das wird an folgendem Programmstück gezeigt:

```
PROGRAM M;
VAR x:REAL;
PROCEDURE A (PA:REAL);
BEGIN
  x:=PA;
END;
BEGIN
  A(1.0)
END.
```

In der Prozedur A ist neben dem formalen Parameter PA auch die globale Variable x bekannt. Sie darf demzufolge in den Anweisungen verwendet werden. Im Beispiel wird ihr der Wert des Parameters zugewiesen.
Der Aufruf A(1.0) läßt nicht erkennen, daß die Hauptprogrammvariable x manipuliert wurde. Die Nützlichkeit solcher „Seiteneffekte" soll hier nicht beurteilt werden. Der Programmierer muß wissen, daß es sie gibt und daß sie sich durch keine Compileroperation ausschalten lassen.

5.5.4. Speicherplatzzuordnung für lokale Variablen

Die durch die Blockstruktur eines PASCAL-Programms festgelegten Gültigkeitsbereiche der Bezeichner führen dazu, daß bei der Programmabarbeitung höchstens die Datenobjekte der aktiven Blöcke, d. h. des Hauptprogramms sowie der gerufenen Prozeduren und Funktionen, manipuliert werden können.
Da Speicherplatz immer eine kostbare Ressource ist, liegt es nahe, nur für Datenobjekte der aktiven Blöcke Speicherplatz bereitzustellen. Weil es sich aber erst zur Programmlaufzeit herausstellt, welche Blöcke aktiv sind, wird diese Strategie als dynamische Speicherplatzzuordnung bezeichnet.
Die Speicherpositionen der Datenobjekte können vom Compiler nur für das Hauptprogramm im voraus berechnet werden.
Die der Prozeduren und Funktionen ergeben sich in Abhängigkeit von ihrer konkreten Aufruffolge. Notwendige Adreßrechnungen müssen bis zur Laufzeit aufgeschoben werden. Die Maschinenbefehlssysteme moderner Rechner unterstützen solche Adreßrechnungen sehr gut, so daß PASCAL-Programme nur geringfügige Laufzeitnachteile gegenüber Programmen mit statischer Speicherplatzzuordnung wie z. B. FORTRAN haben.
Für die Programmierung ergeben sich aus der dynamischen Speicherplatzzuordnung folgende Konsequenzen:

– Der Wert von lokalen Variablen der Prozeduren und Funktionen ist stets als unbestimmt anzusehen, auch falls die Prozedur oder Funktion schon einmal gerufen wurde.
– Rekursive Aufrufe von Prozeduren und Funktionen sind uneingeschränkt möglich.

Das Prinzip wird noch einmal mit Hilfe des Programmrumpfes von Bild 5.31 verdeutlicht. Das Bild 5.33a zeigt den belegten Datenspeicher für die Aufruffolge: Hauptprogramm, Prozedur A, Prozedur B. Beim Verlassen der Prozeduren B und A wird der belegte Datenspeicher wieder freigegeben. Da Prozeduren stets in der umgekehrten Reihenfolge ihres Aufrufs verlassen werden, wird der Datenspeicher stackartig verwaltet. Er wächst zumeist in Richtung fallender Adressen.

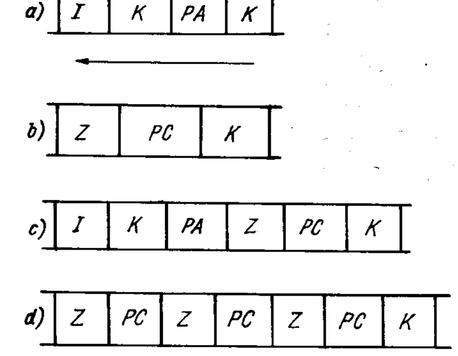


Bild 5.33 Zuordnung von Speicher zu Datenobjekten der aktiven Prozeduren

Die Bilder 5.33b und 5.33c zeigen weitere Situationen. Deutlich tritt auch hervor, daß die Position des Parameters PA und der lokalen Variablen I von der konkreten Aufruffolge abhängt – also erst zur Laufzeit feststeht.
Bild 5.33d zeigt schließlich die Struktur des Datenspeichers bei rekursiven Prozeduraufrufen. C wird vom Hauptprogramm aktiviert und ruft sich dann selbst wieder auf. Bei jedem Betreten von C wird ein neuer Bereich für lokale Variablen angelegt.
TURBO-PASCAL verfährt anders. Der Compiler ordnet den Datenobjekten in der Reihenfolge ihrer Deklaration statisch Platz im Datenspeicherbereich zu. Zur Programmabarbeitungszeit hat jedes Datenobjekt seinen festen Platz im Hauptspeicher. Rekursive Prozeduraufrufe führen so i. allg. zu Fehlern.
Der Vorteil dieser nicht PASCAL-gerechten Strategie liegt darin, daß auch bei Rechnern auf der Basis von 8-Bit-Prozessoren der Zugriff auf Datenobjekte sehr schnell ist und wenig Programmkode erfordert.
Rekursive Prozeduren werden nun in TURBO-PASCAL dadurch möglich, daß beim Betreten einer Prozedur zunächst die „alten" Werte der lokalen Variablen in einem Stack abgelegt und vor Verlassen der Prozedur rückgespeichert werden. Die Erzeugung der dafür erforderlichen Blocktransportbefehle wird durch die Compileroption ○A– und ○A+ ein- bzw. ausgeschaltet.

5.5.5. Vorwärtsdeklaration

Aufgrund des Deklarationszwanges muß ein Bezeichner vor seiner Benutzung deklariert werden. Prozeduren und Funktionen machen hier keine Ausnahme.
Es gibt aber durchaus sinnvolle Programme, bei denen diese Bedingung aufgrund des verwendeten Algorithmus nicht erfüllt werden kann. Das nachfolgende Programmstück verdeutlicht dies:

```
PROCEDURE A(PA:...);
BEGIN
  ...B(x);...
END;
PROCEDURE B(PB:...);
BEGIN
  ...A(y);...
END;
```

Der Aufruf von B im Anweisungsteil von A kann durch den Compiler nicht aufgelöst werden, da bis dorthin B unbekannt ist. Ein Vertauschen der Reihenfolge der Prozeduren A und B führt auch nicht zum Ziel.
Die Lösung erfolgt durch Trennung von Prozedurkopf und dazugehörendem Block mit Hilfe von FORWARD:

```
PROCEDURE B(PB:...);FORWARD;
PROCEDURE A(PA:...);
BEGIN
  ...B(x);...
END;
PROCEDURE B;
BEGIN
  ...A(y);...
END;
```

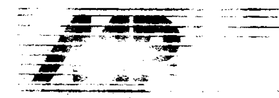
Innerhalb von A kann nun B aufgerufen werden. Durch die vorangegangene FORWARD-Deklaration ist der Kopf von B bekannt. Bei der Angabe des zu B gehörenden Blockes entfällt im Kopf die Parameterliste.

5.5.6. Standardprozeduren und -funktionen

Dem PASCAL-Compiler ist eine Reihe von Prozeduren und Funktionen bereits bekannt. Diese lassen sich in folgende Gruppen einteilen:

1. arithmetische Funktionen
2. Prozeduren und Funktionen zur Ein- und Ausgabe
3. Prozeduren zur Arbeit mit dynamischen Variablen.

Die Behandlung der Gruppen 2 und 3 erfolgt zusammen mit den entsprechenden Datentypen. In diesem Unterpunkt werden die arithmetischen Funktionen besprochen. Es stehen zur Verfügung:



SQR(X)	Quadrat
SQRT(X)	Quadratwurzel
ABS(X)	Absolutwert
SIN(X)	Winkelfunktion
COS(X)	Winkelfunktion
ARCTAN(X)	Winkelfunktion
EXP(X)	e-Funktion
LN(X)	natürlicher Logarithmus

Der aktuelle Parameter kann ein beliebiger Ausdruck vom Typ REAL oder INTEGER sein. Das Resultat ist stets REAL. Eine Ausnahme bildet ABS. Hier hat das Resultat stets den Typ des aktuellen Parameters.
Zur Umwandlung von REAL in INTEGER gibt es die Funktionen

TRUNC(X)	ganzer Teil
ROUND(X)	regelgerechte Rundung.

Die nachfolgenden Standardfunktionen sind auf einfache Typen mit Ausnahme von REAL anwendbar:

ODD(A)	Test auf ungerade
ORD(A)	interne Kodierung von A
SUCC(A)	Nachfolger von A
PRED(A)	Vorgänger von A
CHR(I)	Konstante vom Typ CHAR mit interner Kodierung I

Die Funktion CHR kann vorteilhaft zur Konstruktion von CHAR-Konstanten für nicht darstellbare Zeichen des ASCII-Kodes genutzt werden. Zum Beispiel liefert CHR(27) den Kode für Escape.
Die Wirkung der Standardfunktionen ORD, SUCC und PRED wird durch die Anwendung auf den Aufzählungstyp

```
TYPE T = (Mo,Di,Mi,Don,Fr,Sa,So);
VAR X : T;
```

gezeigt:

```
ORD(Mo) liefert: 0
ORD(Di) liefert: 1
.
.
.
ORD(So) liefert: 6
```

Die Anwendung von Nachfolger- und Vorgängerfunktionen liefern:

```
SUCC(Mo) liefert: Di
SUCC(Di) liefert: Mi
.
.
.
SUCC(Sa) liefert: So

PRED(So) liefert: Sa
PRED(Sa) liefert: Fr
.
.
.
PRED(Di) liefert: Mi
```

Nicht definiert sind die Funktionswerte SUCC(So) und PRED(Mo).
Darüber hinaus stellen viele PASCAL-Systeme weitere Standardprozeduren und -funktionen bereit, mit denen Eigenschaften der Gerätetechnik oder des zugrunde liegenden Betriebssystems auf das Niveau der Sprache gehoben werden. In TURBO-PASCAL gibt es davon mehr als 50. Eine Auswahl wird hier angegeben:

ADDR(X)	Adresse der Variablen X
SIZEOF(X)	Speicherplatzbedarf von X in Byte
LO(X)	Niederwertiges Byte von X
HI(X)	Höherwertiges Byte von X
SWAP(X)	Vertauschen der Bytes von X
MOVE(VON,ZU,ANZ)	Blocktransport von "ANZ"-Bytes
BIOS(I,J)	BIOS-Ruf I mit Parameter J
BDOS(I,J)	BDOS-Ruf I mit Parameter J
HALT	Abbruch der Programmabarbeitung

6. Datentyp File, Ein- und Ausgabeorganisation

6.1. Einführung

Der Datentyp File bildet in PASCAL die Grundlage für die Ein- und Ausgabe von Daten von bzw. zu den peripheren Speichermedien.
Ein Datenobjekt vom Filetyp ist eine Folge von Komponenten gleichen Typs. Der Typ einer Komponente ist beliebig. Der Zugriff zu den Komponenten ist nur sequentiell möglich.
Die Brücke zu den peripheren Geräten wird geschlagen, indem Variablen vom Filetyp als Files im Sinne eines Betriebssystems angelegt werden.

6.2. Deklaration und Zugriff

Die Deklaration von Filetypen ist eine Alternative im Syntaxdiagramm "typ" (Bild 3.1). Sie wird im Bild 6.1 gezeigt. Die reservierten Bezeichner FILE und OF leiten die Deklaration ein. Anschließend wird der Komponententyp angegeben. Jeder Filevariablen wird eine Puffervariable zugeordnet. Sie wird gebildet durch Anfügen des Zeichens "^" an den Bezeichner einer Filevariablen.



Bild 6.1 Syntaxdiagramm „filetyp"

Über die Puffervariable kann auf die sogenannte aktuelle Komponente eines Files zugegriffen werden. Sie läßt sich wie eine Variable verwenden.
Es folgen einige Beispiele:

a) Komponententyp INTEGER:
```
VAR F: FILE OF INTEGER;
    I: INTEGER;
.
F^:=1; I:=2*F^+1; ...
```
b) Komponententyp REAL
```
VAR F: FILE OF REAL;
    X: REAL;
X:=F^*2.5; F^:=0.5; ...
```
c) Komponente vom Aufzählungstyp
```
TYPE T=(Mo,Di,Mi,Don,Fr,Sa,So);
VAR F: FILE OF T;
    Z:T;
.
.
.
F^:=Di; Z:=F^; ...
```

F^ ist die Puffervariable. Durch die Zugriffe wird die aktuelle Komponente nicht verstellt. Dazu gibt es Standardprozeduren, die im nächsten Unterpunkt besprochen werden.

6.3. Standardprozeduren und -funktionen für den Datentyp File

Die Ein- und Ausgabe erfordert einige komplexe Leistungen. Sie werden in Form von Standardprozeduren und -funktionen bereitgestellt. In der nachfolgenden Aufstellung bezeichnet f eine Variable vom Filetyp:

RESET(f) REWRITE(f)	Vorbereiten des Files für Lesen bzw. Schreiben. Die Puffervariable wird auf die erste Komponente positioniert.
GET(f) PUT(f)	Weiterrücken der Puffervariablen auf die nächste Komponente des Eingabe- bzw. Ausgabefiles
EOF(f)	Funktion, liefert den Wert TRUE, falls Fileende erreicht ist.

Bei Eingabefiles zeigt EOF(f)=TRUE an, daß die Puffervariable auf Fileende positioniert wurde und keine weiteren Komponenten vorhanden sind. Der Zugriff über die Puffervariable f^ liefert einen unbestimmten Wert.
Für Ausgabefiles zeigt EOF(f)=TRUE an, daß die Puffervariable auf das Fileende positioniert ist und die nächste Komponente angefügt werden kann.
Mit diesen fünf Prozeduren kann die gesamte Ein- und Ausgabe durchgeführt werden. Später werden weitere Prozeduren besprochen, die die Arbeit erleichtern, aber keine prinzipiell neuen Leistungen erbringen.
Es gibt zwei Grundaufgaben bei der Ein- und Ausgabe: das Schreiben und das Lesen eines Files. Die programmtechnische Lösung wird jetzt gezeigt. Sie ist unabhängig vom Typ der Komponenten. Er wird deshalb auch offen gelassen:

```
PROGRAM Schreiben;
TYPE Komp = ...;
VAR F:FILE OF Komp;
    X:Komp;
    I:INTEGER;
BEGIN
  REWRITE(F);
  FOR I:=1 TO 100 DO BEGIN
    F^:=X; PUT(F)
  END
END.
```

```
PROGRAM Lesen;
TYPE Komp = ...;
VAR F:FILE OF Komp;
    X:Komp;
BEGIN
  WHILE NOT EOF(F) DO BEGIN
  X:=F^; GET(F)
  END
END.
```

Natürlich hätte das Lesen auch innerhalb einer FOR-Anweisung erfolgen können. Die gezeigte Variante hat aber den Vorteil, daß sie auch funktioniert, wenn die Anzahl der Komponenten unbekannt ist.
Beim Lesen findet keine Überprüfung des Komponententyps statt. Weiterhin erlaubt Standard-PASCAL für ein File nur Lese- oder Schreibzugriffe. Das Verändern eines Files muß durch Kopieren gelöst werden.
Durch zwei zusätzliche Standardprozeduren wird die Arbeit mit Filevariablen weiter vereinfacht:

READ(f,v) entspricht v:=f^; GET(f)
WRITE(f,a) entspricht f^:=a; PUT(f).

Hier stehen v für eine Variable und a für einen Ausdruck vom Komponententyp der Filevariablen f.
Als weitere Vereinfachung sind Listen von Variablen bzw. Ausdrücken erlaubt.:

READ(f,v_1,v_2, ...)
WRITE(f,a_1,a_2, ...).

Unglücklicherweise weist TURBO-PASCAL einige Abweichungen auf:

1. Die Prozeduren PUT und GET gibt es nicht.
2. Auf die Puffervariable kann nicht Bezug genommen werden.
3. Mit der Standardprozedur ASSIGN(f,a) muß der Filevariablen f eine Datei zugeordnet werden. Der Wert des Ausdrucks a muß eine Zeichenkette sein.
4. Nach Abschluß der Arbeit ist insbesondere für Ausgabefiles durch Aufruf der Standardprozedur CLOSE(f) das Leeren der Puffer zu organisieren.

Die für TURBO-PASCAL modifizierte Lösung der Grundaufgaben wird jetzt gezeigt. Zuerst wieder das Schreiben auf die Datei A:TEST.DAT:

```
...
BEGIN
  ASSIGN(F,'A:TEST.DAT');
  REWRITE(F);
  FOR I:=1 TO 100 DO WRITE(F,X);
  CLOSE(F)
END.
```

Und das Lesen:

```
...
BEGIN
  ASSIGN(F,'A:TEST.DAT');
  RESET(F);
  WHILE NOT EOF(F) DO READ(F,X)
END.
```

Der Typ der Komponenten kann auch hier beliebig sein.
Treten Fehler während der Ein- oder Ausgabe auf, wird in der Regel die Abarbeitung des PASCAL-Programms abgebrochen und eine Fehlermeldung ausgegeben.
TURBO-PASCAL ermöglicht die Behandlung von Ein- und Ausgabefehlern auf dem Niveau von PASCAL. Durch Aufruf der Standardfunktion IORESULT wird der Status der zuletzt durchgeführten Ein- oder Ausgabe zurückgegeben. Eine Auswahl der wichtigsten Werte folgt:

000	Kein Fehler
001	File exisitiert nicht
002	File nicht für Eingabe eröffnet
003	File nicht für Ausgabe eröffnet
004	File nicht eröffnet
091	Positionieren über Fileende
240	Disk-Schreibfehler
241	Directory voll
242	File zu groß
255	File verloren

Damit die Ausführung nutzereigener Fehlermaßnahmen möglich ist, müssen die Standardmaßnahmen des PASCAL-Systems unterdrückt werden.
Das ist mit der Compileroption ○I– möglich. Die Standardmaßnahmen werden durch ○I+ wieder aktiviert.

6.5. Textfiles

In nahezu allen EDV-Projekten spielt die Verarbeitung von Erfassungsbelegen, Drucklisten und Bildschirmmenüs eine wichtige Rolle. In PASCAL können diese Datenstrukturen durch

TYPE TEXT = FILE OF CHAR

beschrieben werden.
Das Problem besteht nun darin, daß Texte in Zeilen und Seiten strukturiert sind und daß die Repräsentation dieser Zeilen und Seiten vom jeweiligen Rechner und Betriebssystem abhängen. Aus diesem Grunde wird die Zeilen- und Seitenstruktur durch Standardprozeduren auf das Niveau von PASCAL gehoben:

READLN(f)	Positionierung der Puffervariablen auf den nächsten Zeilenanfang
WRITELN(f)	Zeilenabschluß im Ausgabefile
EOLN(f)	Funktion, liefert den Wert TRUE, falls in einem Eingabefile das Zeilenende erreicht ist
PAGE(f)	Beginn einer neuen Seite im Ausgabefile.

Auch hier gibt es wieder die Vereinfachungen:

READLN(f,v1,v2,...)
für READ(f,v1,v2,...); READLN(f)
WRITELN(f,a1,a2,...)
für WRITE(f,a1,a2,...); WRITELN(f)

Die Grundaufgabe bei der Arbeit mit Textfiles besteht im Erkennen und Erzeugen der Zeilenstruktur. Die programmtechnische Lösung wird im Bild 6.2 gezeigt. Den Kern des Programms bilden zwei ineinandergeschachtelte While-Anweisungen. Mit der inneren wird eine Zeile von INFILE auf OUTFILE kopiert. Die While-Anweisung wird am Zeilenende wegen EOLN(INFILE)=TRUE verlassen. Durch Aufruf von READLN(INFILE) muß nun auf die nächste Zeile im Eingabefile vorgesetzt werden. Mit WRITELN(OUTFILE) wird die Zeile im Ausgabefile abgeschlossen. Die äußere While-Schleife sorgt nun dafür, daß das Zeilenkopieren solange durchgeführt wird, bis das Fileende des Eingabefiles erreicht ist.

```
PROGRAM COPY;
VAR C: CHAR;
    INFILE,OUTFILE: TEXT;
BEGIN
  ASSIGN(INFILE,...); RESET(INFILE);
  ASSIGN(OUTFILE,...); REWRITE(OUTFILE);
  WHILE NOT EOF(INFILE) DO BEGIN
    WHILE NOT EOLN(INFILE) DO BEGIN
      READ(INFILE,C); WRITE(OUTFILE,C)
    END;
    READLN(INFILE); WRITELN(OUTFILE)
  END;
  CLOSE(OUTFILE)
END.
```

Bild 6.2 Kopierprogramm für Textfiles

Die PASCAL-Systeme vereinfachen die Arbeit mit Textfiles weiter. Für die Datentypen INTEGER, REAL und BOOLEAN werden Konvertierungen durchgeführt. Bei WRITE sind als aktuelle Parameter Zeichenketten zugelassen. So ist folgendes Programmstück korrekt:

```
VAR INFILE,OUTFILE: TEXT;
    I: INTEGER; B: BOOLEAN;
    X: REAL;
    ...
READ(INFILE,I,B,X);
WRITE(OUTFILE,3*I,B,'X:= ',X);
```

Im Eingabefile wirken alle Nichtziffern als Begrenzer für Zahlen. Beliebige Bezeichner, die mit den Buchstaben F oder T beginnen, werden als FALSE oder TRUE interpretiert.
Bei WRITE können die Ausdrücke mit Angaben zur Formatgestaltung versehen werden. Sie haben die Form:

ausdruck:m oder
ausdruck:m:n.

(wird fortgesetzt)

Ein- und Mehrprozessorsysteme mit Multibus-Architektur

Dr. Hans-Joachim Schwertfeger,
Wolfgang Krüger
VEB Robotron Elektronik Dresden

Für die multivalente Nutzung eines Mikroprozessorsystems ist eine wesentliche Voraussetzung, daß seine Schnittstellen der Umwelt gegenüber weitgehend standardisiert sind. Vermittelndes Glied zwischen System und Umwelt bildet im allgemeinen das Bussystem, für welches sich in der Vergangenheit als ein international anerkannter Standard der von Intel geschaffene Multibus herausgestellt hat. Die Entwicklung der Hardware zu größeren Verarbeitungsbreiten und die verstärkte Verwendung von Multiprozessorsystemen erforderten eine Weiterentwicklung dieses Buskonzepts, die vom Multibus I zum Multibus II führte; dessen Darstellung und der Vergleich mit seinem Vorgänger sind Inhalt der folgenden Ausführungen.

1. Multibus-Architektur

1.1. Komponenten des Bussystems

Die Bussysteme des Multibus I und II bestehen aus den Teilen
– Systembus
– Lokalbus
CPU-Bus
Speicherbus
E/A-Bus
– Zusatzbusse
lokaler E/A-Bus
DMA-Bus.
Diese Dreiteilung ist dem Multibus I und II gemeinsam; jedoch existieren wesentliche Unterschiede hinsichtlich der Struktur und Arbeitsweise von System- und Lokalbus. Die Zusatzbusse besitzen die gleiche Ausführung:
– Lokaler E/A-Bus (iSBX)
Erweiterung von CPU- und intelligenten E/A-Steckeinheiten um E/A-Geräte geringer Komplexität (serielle und parallele Schnittstellen, Floppy-Disk-Steuergeräte). Das bedeutet effektive Erweiterung des Funktionsumfanges und Entlastung des Systembusses.
– DMA-Bus
Anschluß von transferintensiven E/A-Geräten über größere Entfernungen (mehrere Meter).
Die Aufteilung in System- und Lokalbus ist zur Erhöhung der Leistungsfähigkeit eines Multimikroprozessorsystems (MMPS) durch Entlastung des Systembusses unbedingt notwendig.

1.2. Grundstruktur eines MMPS

Ein MMPS (Bild 1) unter Verwendung des Multibus besteht aus Prozessor- und Speichermoduln. Als Prozessormoduln sind solche mit Verarbeitungs- und solche mit Ein- und Ausgabesteuerfunktionen verwendbar. Diesen Prozessormoduln können Speichereinheiten und E/A-Geräte je nach Erfordernissen zugeordnet werden.

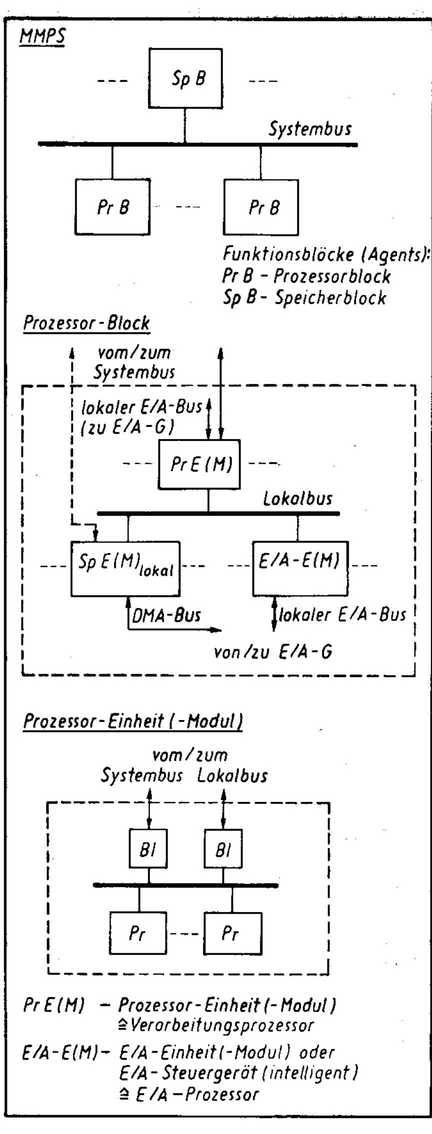


Bild 1 Grundstruktur eines MMPS auf Multibusbasis

Die entsprechend den Funktions- oder Einwirkungsbereichen zusammengefaßten Prozessor- und Speichermoduln bilden jeweils einen Funktionsblock (FB, Agent), der als physikalische Einheit ein Interface zum Systembus (öffentlicher Systembus), dem zentralen Bindeglied des MMPS, besitzt. Innerhalb jedes Funktionsblocks existieren Funktionseinheiten (FE) mit Verarbeitungs- und/oder intelligenten Ein-/Ausgabefunktionen sowie Speicher, die über einen Lokalbus (privater Systembus) zusammengeschaltet sind. Mit Hilfe von Zusatzbussen sind beliebige nicht intelligente E/A-Geräte anschaltbar.

1.3. Struktur und Arbeitsprinzipien

Zwischen den an einem Bus oder mehreren Bussen mit u. U. unterschiedlicher Architektur arbeitenden Funktionseinheiten oder Moduln ist ein Informationsaustausch zu verwirklichen. Er kann auf verschiedene Art und Weise organisiert werden, und zwar
– mit Steuerung durch Interrupts und direktem Datentransfer
– mit Nutzung globaler Speicherbereiche (als Puffer) mit Steuerung durch Interrupts
– mit Nutzung individueller Speicherbereiche (als Puffer) ohne Verwendung von Interrupts.
Im ersten Fall erfolgt der Informationsaustausch auf der Basis einer Unterbrechungssteuerung, im dritten Fall unter Verwendung eines Kommunikationsprotokolls ohne Quittung. Der zweite Fall entspricht einer Kombination beider Verfahren.
Die Bilder 2 und 3 stellen Varianten mit Steuerung des Informationsaustauschs über Interrupt dar. Kommuniziert ein intelligenter mit einem nichtintelligenten Modul, ist diese Steuerung in einer Richtung, im Falle zweier intelligenter Moduln in beiden Richtungen zu verwirklichen. Diese Art der Kommunikation stellt das niedrigste Niveau dar. Im Bild 2 werden Kommandos oder Daten direkt vom Prozessor zur E/A-Einheit übertragen, für die Übertragung in umgekehrter Richtung ist eine Vorbereitung mit Hilfe eines Interrupts erforderlich. Dieser wird gesendet, wenn Daten oder Status verfügbar sind. Im Bild 3 ist in beiden Richtungen eine Vorbereitung durch Interrupt erforderlich. Auf Grund von dessen Ausführung können anschließend Befehle, Daten oder Status übernommen werden.
Bild 4 stellt den Informationsaustausch mittels gemeinsamer Speicherbereiche für die Zwischenspeicherung der zu übertragenden Informationen dar. Diese werden nach Verfügbarkeit in die Speicherbereiche ohne Kenntnis des Zustands der intelligenten Ziel-FE übertragen. Die Weiterleitung der Information erfolgt nach Vorbereitung der Ziel-FE mit Interrupt und dessen Ausführung.

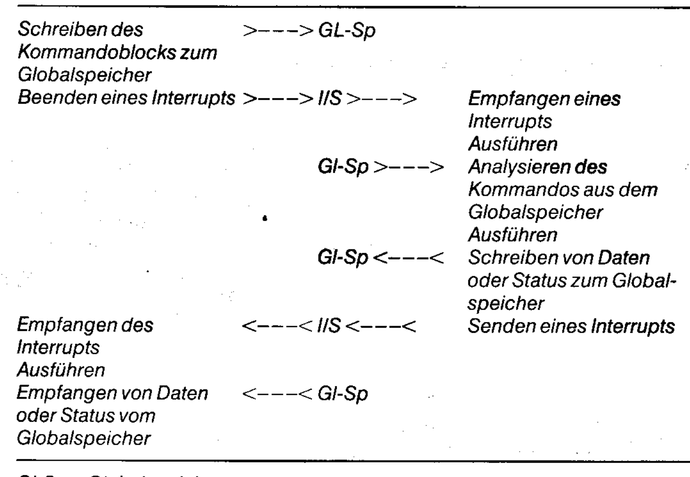
Prozessor		*E/A-Gerät*
Senden von Befehl oder Daten	>––––> I/S >––––>	*Empfangen von Befehl oder Daten Ausführen Daten oder Status verfügbar*
Empfangen eines Interrupts Ausführen	<––––< I/S <––––<	*Senden eines Interrupts*
Empfangen von Daten oder Status	<––––< I/S <––––<	*Senden von Daten oder Status*

I/S – Interface-Schnittstelle

Bild 2 Informationsaustausch zwischen Prozessor und nichtintelligentem E/A-Gerät unter Verwendung von Interrupts

Prozessor		*E/A-Steuergerät*
Schreiben eines Kommandoblocks zum E/A-Steuergerät	>––––> I/S >––––>	*Kommando verursacht Interrupt Ausführen Daten oder Status verfügbar*
Empfangen eines Interrupts Ausführen	<––––< I/S <––––<	*Senden eines Interrupts*
Empfangen von Daten oder Status	<––––< I/S <––––<	*Senden von Daten oder Status*

Bild 3 Informationsaustausch zwischen Prozessor und intelligentem E/A-Steuergerät unter Verwendung von Interrupts



Gl-Sp – Globalspeicher

Bild 4 Informationsaustausch zwischen Prozessor und intelligentem E/A-Steuergerät unter Verwendung von Interrupts und gemeinsamer Speicherbereiche für den Informationsaustausch

Prozessor — *E/A-Steuergerät*

lange Nachricht >--> L-Sp
(solicited)
Pufferanforderung
ISt >--> I/S >--> ISt
--> Puffer notwendig
<-- Puffer bereit
Pufferbereitstellung
ISt <--< I/S <--< ISt
Daten
>--> >-->
L-Sp | | L-Sp lange Nachricht
>--> >--> (solicited)
Transfer beendet Transfer beendet
<--< >-->

Bild 6 Informationsaustausch zwischen Prozessor und intelligentem E/A-Steuergerät ohne Verwendung von Interrupts mit individuellen Speicherbereichen für den Informationsaustausch (Mehr-Block-Übertragung)

Prozessor — *E/A-Steuergerät*

kurze Nachricht >--> L-Sp
(unsolicited)
ISt >--> I/S >--> ISt
L-Sp >--> kurze Nachricht
(unsolicited)
Ausführung

L-Sp – Lokalspeicher
ISt – Interfacesteuerung

Bild 5 Informationsaustausch zwischen Prozessor und intelligentem E/A-Steuergerät ohne Verwendung von Interrupts mit individuellen Speicherbereichen für den Informationsaustausch (Ein-Block-Übertragung)

Bus-Konfiguration II Prozessor				*Bus-Konfiguration I E/A-Steuergerät*
Schreiben des Kommandoblocks zum E/A-St.-Gerät	*Link >--> K II>K I*	*>-->*	*I/S >------>*	*Kommando verursacht Interrupt*
				Ausführen Daten oder Status verfügbar
Empfangen des Interrupts	*<------<*	*Link I/S <--< K II<K I*	*<--<*	*Senden eines Interrupts*
Lesen von Daten oder Status aus Gl-Sp	*Link >--> K II>K I*	*>-->*	*I/S >------>*	*Slave liefert Daten oder Status*
Schreibkommando zur Löschung des Interrupts	*Link >--> K II>K I*	*>-->*	*I/S >------>*	*Kommando verursacht Interrupt*
				Ausführung: Löschung des Interrupts

Link K II<>K I – Umsetzung von K II auf K I und umgekehrt

Bild 7 Informationsübertragung zwischen einem Prozessor der Bus-Konfiguration II und einem intelligenten E/A-Steuergerät der Buskonfiguration I (Übertragung von kurzen Nachrichten)

Die Bilder 5 und 6 charakterisieren eine Ein-Block- bzw. Mehr-Block-Übertragung mit Hilfe eines Kommunikationsprotokolls unter Verwendung von individuellen Speicherbereichen (interne Arbeitsspeicher als spezieller Nachrichtenraum) ohne Nutzung von Interrupts. Das jeweils übertragene Nachrichtenpaket enthält u. a. Ziel- und Quelladresse der am Informationsaustausch beteiligten FE bzw. Moduln (ohne Kenntnis der Adressen im internen Speicher, deren Zuordnung durch den entsprechenden intelligenten Modul organisiert wird). Durch eine derartige Organisation ist es möglich, unter bestimmten Betriebsbedingungen eine wesentliche Erhöhung der Übertragungsrate zu erzielen. Auf diese Weise ist auch eine virtuelle Interruptsignalisierung realisierbar.
Bild 7 zeigt die Möglichkeit des Informationsaustausches zwischen Moduln, die in voneinander abweichenden Bus-Architekturen arbeiten. In diesem Fall ist bei unterschiedlichen Organisationsmethoden eine Anpassung in Form von Busumsetzern für beide Übertragungsrichtungen notwendig. Sie ermöglicht den Übergang vom Steuerprinzip auf der Basis eines Kommunikationsprotokolls mit Verwendung eines Nachrichtenadreßraumes zum Steuerprinzip mit Interrupt und umgekehrt. Eine solche Kommunikation ist in drei Phasen zu unterteilen: Vorbereitung, Durchführung, Beendigung.
Im dargestellten Fall handelt es sich um die Kommunikation zwischen zwei intelligenten Moduln. Es ergibt sich eine Kombination zwischen den beiden genannten Mechanismen. Der Inhalt des Speicher- oder E/A-Raums der Buskonfiguration II wird direkt in den entsprechenden Raum der Buskonfiguration I geschrieben. Nach Auswertung des Kommandoblocks, Ausführung des Kommandos und Schreiben von Daten und/oder Status in den Globalspeicher wird ein Interrupt vom Ziel-Modul erzeugt und zum Quell-Modul übertragen. Daran anschließend erfolgt die Übernahme von Daten und/oder Status durch den Zielmodul aus dem Globalspeicher. Der Vorgang wird durch Löschen des Interrupts im Ziel-Modul durch den Quell-Modul abgeschlossen.

2. Systemkonfiguration mit Multibus I

Die Multibus-I-Architektur besteht aus einem öffentlichen und einem privaten Systembus, also jeweils einem Bus mit globalem und lokalem Charakter (Bild 8). Der öffentliche Systembus (Bild 8a) dient dem Informationsaustausch zwischen Prozessormoduln und öffentlichen oder Global-Speichern sowie E/A-Einheiten. Bei Vorhandensein mehrerer Prozessormoduln ist von jedem der Zugriff zu allen öffentlichen Speicher- und E/A-Einheiten möglich. In diesem Fall erfolgt die Buszuweisung an einen dieser Prozessormoduln auf der Basis eines verteilten Zuweisungs- bzw. Zuteilungsmechanismus mit einem Bus-Arbiter je Prozessormodul. Eine Übersicht gibt Tafel 1. Über den privaten Systembus (Bild 8b) ist ein Prozessormodul mit den lokalen Ressourcen, d. h. Speicher und E/A-Einheiten, verbunden. Zu diesen lokalen Ressourcen ist nur vom zugeordneten Prozessormodul der Zugriff möglich. Der Prozessormodul (Bild 8c) ist als einfacher und komplexer Modul realisierbar. Ein einfacher Prozessormodul besteht aus einem (Verarbeitungs- oder Ein-/Ausgabe-)Prozessor und je einem Businterface zum privaten und öffentlichen Systembus. Ein komplexer Prozessormodul besteht aus mehreren Prozessoren mit Verarbeitungs- und/oder Ein-Ausgabeaufgaben und den genannten Businterfaces mit Sender/Empfänger und Bussteuereinheit. Die Baugruppen des Prozessormoduls arbeiten über einen Lokalbus zusammen.

3. Systemkonfigurationen mit Multibus II

3.1. Allgemeiner Aufbau

Hauptbestandteile der Multibus-II-Architektur sind System- und Lokalbus.
Der Systembus (Bild 9a) dient als zentrales Verbindungsglied dem Informationsaustausch zwischen Prozessorbereichen und, wenn erforderlich, zwischen Prozessorbereichen und globalen Speicher- und E/A-Moduln. Er besteht aus zwei voneinander unabhängigen Bussen:

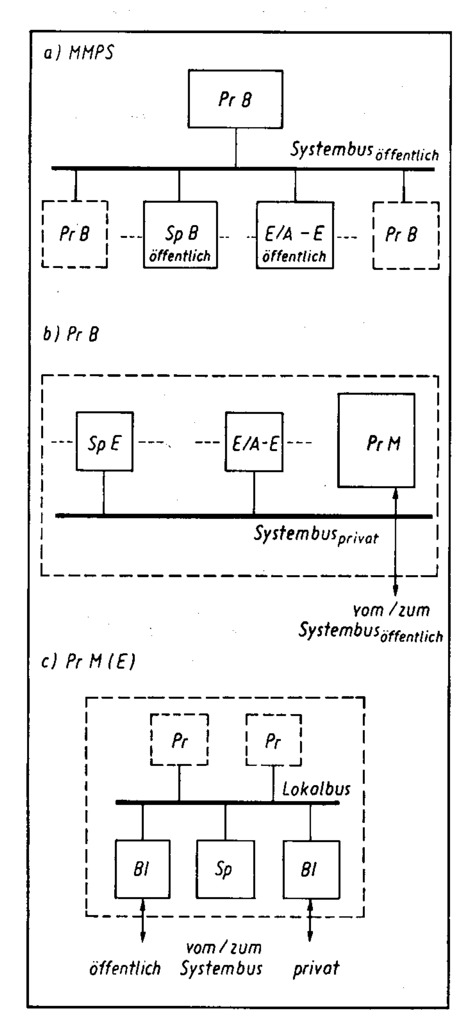


Bild 8 Grundstruktur eines MMPS auf Multibus-I-Basis

Tafel 1 Übersicht über Lokal- und Systembus bei Multibus I

Lokalbus (≙ internem Bus des Prozessormoduls in Bild 1):

– Zusammenschalten/-wirken von Prozessoren (Verarbeitungs-Ein-/Ausgabeprozessoren)
– Zusammenwirken mit anderen SK über Businterface-Baustein (Steuer- und Puffereinheiten) Latch Sender/Empfänger

Systembus:

– öffentlicher Systembus (Globalbus; ≙ Systembus in Bild 1)
Zusammenarbeit von Prozessormoduln mit gemeinsamen Speicher- und Ein- und Ausgabe-Moduln
– privater Systembus (Lokalbus)
Zusammenarbeit von Prozessormoduln am Lokalbus (≙ Lokalbus im Prozessor-Block in Bild 1) mit individuellen Speichereinheiten (über Resident-Bus) und Ein-/Ausgabeeinheiten (privater/individueller/lokaler E/A-Bus)

– Der parallele Systembus (iPSB) ist ein 32 Bit breiter Verbindungsweg für die Prozessormoduln eines Systems; er übernimmt die Arbitrierung und ermöglicht den Zugriff auf globale Speicher- und E/A-Ressourcen.
– Der serielle Systembus (iSSB) verbindet Moduln eines Systems mit Hilfe einer Zweidrahtleitung über größere Entfernungen miteinander und erlaubt dadurch die räumlich getrennte Anordnung von Systemkomponenten. Der Lokalbus (Bild 9b) ermöglicht die Verbindung zwischen Prozessor- und intelligenten E/A-FE und lokalen Speichereinheiten. Die Gesamtheit der an einen Lokalbus angeschalteten Baugruppen wird als Funktionsblock (Agent) bezeichnet. Auf Grund der lokalen Zuordnung von Prozessor und Speicher und der Organisation auf der Basis einer überlappten Arbeitsweise (pipelining) von Adreß- und Datenphasen sind eine Entlastung des Systembusses und eine Leistungserhöhung möglich. Innerhalb eines FB können maximal 6 FE oder Moduln, davon bis zu zwei FE mit Masterfunktion (Prozessoren mit Verarbeitungs- und/oder E/A-Funktion), eingesetzt werden. Die lokalen Speichereinheiten sind zusätzlich an den parallelen Systembus anschaltbar, zu ihnen kann dann von Prozessor-FE anderer FB zugegriffen werden.
Die über einen parallelen Systembus zusammenarbeitenden Prozessorbereiche bilden einen Multibus-Bereich (Bild 9c). Mehrere Multibus-Bereiche werden über einen seriellen Systembus zusammengeschaltet (Bild 9d).

3.2. Komponenten des Multibus-II-Systems

3.2.1. Paralleler Systembus iSSB

3.2.1.1. Arbeitsweise von Daten- und Adreßbus

Der parallele Systembus ist die zentrale Kommunikationsverbindung in jedem Multibus-Bereich. Er arbeitet synchron mit einer Taktfrequenz von 10 MHz. Die synchrone Arbeitsweise hat folgende Vorteile:
– Erhöhung von Zuverlässigkeit und Störsicherheit
– Vereinfachung der Steuerung mit Verringerung der Verzögerungszeiten der Signale und damit verbundene Erhöhung der Arbeitsgeschwindigkeit.
Daten- und Adreßbus arbeiten multiplex und sind jeweils maximal 32 Bit breit; andere mögliche Formate sind 8, 16 und 24 Bit. Es besteht daher die Möglichkeit, 8-, 16- und 32-Bit-Mikroprozessoren an den Bus zu schalten. Kommando- und Statussignale werden ebenfalls im Multiplexbereich behandelt. Daraus resultiert ein geringerer Aufwand an Leitungen, Schaltkreisen und Stromverbrauch.
Zwei Transfer-Modi sind möglich:
– In einem Einfach-Zyklus werden im ersten Takt die Adresse und während des folgenden Taktes die Daten übertragen. Der Durchsatz beträgt 20 MByte/s.
– In einem Mehrfachzyklus werden im ersten Takt die Adresse und während der folgenden Takte ein Datenblock beliebiger Länge übertragen. Der Durchsatz beträgt 40 MByte/s. Dieser Modus ist besonders für den Betrieb von 32-Bit-Mikrorechnern zweckmäßig.
Zur Erhöhung der Störsicherheit sind jeweils 8 Adreß-/Datenleitungen durch ein Paritätsbit geschützt, wodurch eine sofortige Fehlererkennung möglich ist.

3.2.1.2. Bedeutung von Kommando- und Statusleitungen

Die Kommando- und Statusleitungen der System-Control-Gruppe (SC 0...7) haben zu unterschiedlichen Zeitpunkten innerhalb des Buszyklus verschiedene Bedeutung; sie sind durch ein Paritätsbit (SC 8) geschützt (siehe Tafel 2).
Weitere Fehlermöglichkeiten, wie BUSERR, TIMEOUT, NACK, führen zu einem sofortigen Abbruch des Buszyklus und Übergang in einen Fehlerzyklus.

3.2.1.3. Arbitrierung und Interruptsteuerung

Die Multibus-II-Architektur ist durch eine dezentrale Arbitrierung gekennzeichnet; jeder intelligente FB ist an diesem Vorgang beteiligt. Die Prioritätsauflösung erfolgt in maximal drei Takten, wird aber überlappt mit laufenden Transferzyklen durchgeführt, so daß minimale Zeitverluste, insbesondere bei hoher Busbelastung, entstehen. Wenn mehrere Busmaster gleichzeitig zum Bus zugreifen (BREQ), sorgt die verteilte Arbitrierlogik für die Zugriffsberechtigung des FB (der FE) mit der bei der Initialisierung festgelegten höchsten Priorität. Alle in einem Prioritätszyklus vorliegenden und bewerteten Prioritäten werden bearbeitet, ehe ein neuer Arbitrierzyklus durchgeführt wird. Dadurch wird erreicht, daß eine FE mit hoher Priorität den Bus nicht zu hoch auslastet und dadurch FEs mit niedriger Priorität nicht mehr arbeiten können. Es besteht aber die Möglichkeit, diesen Algorithmus mit einem High-Priority-Request-Signal (ARB 5) zu unterbrechen und einen High-Priority-Zyklus einzuschieben. Wenn mehrere FEs dieses Anforderungssignal ausgeben, kommt der beschriebene Prioritätsmechanismus zur Anwendung.

Tafel 2 Bedeutung der Kommando- und Statusleitungen

Adreßphase		*Datentransferphase*	
SC 0	*Beginn eines Buszyklus = ALE-Signal (address latch enable)*		
SC 1	*Sperrung des Busses gegen Zugriff anderer Moduln zum Zwecke der Ausführung mehrerer Buszyklen*		
SC 2)	*Spezifikation der Breite*	*SC 2*	*Ende des Buszyklus H: Datentransfer*
SC 3)	*des Datentransfers*		*L: Ende des Datentransfers mit einem nachfolgenden Datentakt (Steuerung des Blocktransfers)*
		SC 3) SC 4)	*Implementierung eines zweiseitigen Handshake-Protokolls SC 3: Verzögerung der Daten durch den Sendemodul Requester ready (RQRDY)*
SC 4)	*Definition des Adreßraums*		*SC 4: Verzögerung der Daten durch den Empfangsmodul Replier ready (RPRDY)*
SC 5)	*(Speicher-, E/A, Interconnect- oder Messageadreßraum*	*SC 5) SC 6) SC 7)*	*Rückmeldung Empfänger Sender bei Fehler (u. a. illegale Adresse, Ende des Speicherbereichs bei Blocktransfer, Paritätsfehler) Meldung am Ende des Buszyklus*
SC 6	*Kennzeichnung von Lese- und Schreibzugriff*		

a) MMPS

Pr B

iSSB
iPSB

Sp $E_{gl.}$ --- E/A - $E_{gl.}$

b) Pr B

iSBX - Bus
iSBX - Bus
Multichannel - Bus
VPr --- Sp --- E/A - SG
iLBX II - Bus
iPSB - Bus
iSSB - Bus

c) Zusammenwirken in einem Multibus - Bereich

iSBX - Bus
iSBX - Bus
VPr Sp
iLBX II - Bus
VPr Sp
iLBX II - Bus
EA - SG
iPSB - Bus

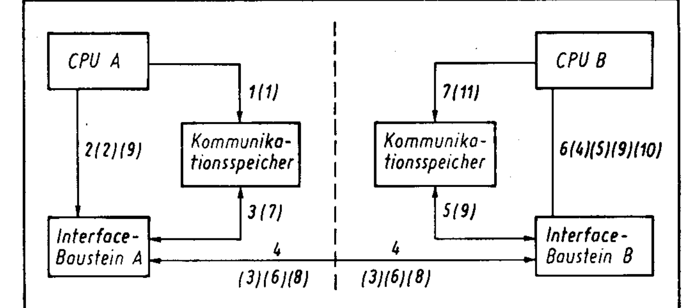
d) Zusammenwirken mehrerer Multibus - Bereiche

MULTIBUS - BEREICH
iSBX-Bus Pr B
iSBX-Bus
VPr Sp E/A-SG
iLBXII -Bus
iPSB - Bus

iSSB - Bus - serieller Systembus
iLBX II - Bus - lokaler Bus
iSBX - Bus - lokaler E/A - Bus
Multichannel - Bus - DMA - Bus

MULTIBUS - BEREICH
iSBX - Bus Pr B
VPr Sp E/A - SG
iLBX II - Bus
iPSB - Bus
iSSB - Bus

iPSB - Bus - paralleler Systembus
E/A - SG - E/A - Steuergerät (- Prozessor)
E/A - G - E/A - Gerät
MMPS auf Multibus - II - Basis



Ablauf bei „unsdicited" - Übertragung

1 Ablage der zu übertragenden Nachricht im Arbeitsspeicher durch die CPU A
2 Information des Interface-Bausteins A über Forderung einer Informationsübertragung
3-5 Senden und Empfangen der Nachricht (Informationsübertragung)
6 Information der CPU B durch den Interface-Baustein B
7 Übernahme der übertragenen Nachricht aus dem Arbeitsspeicher zur CPU B

Ablauf bei „solicited" - Übertragung

1 wie oben
2 wie oben
3 Übermittlung der Kommunikationsspeicher-Forderung
4-5 Anforderung und Bereitstellung des Kommunikationsspeichers durch die CPU B
6 Information über Kommunikationsspeicher-Bereitstellung
7-9 Senden und Empfangen der Nachricht (Informationsübertragung)
10 Information der CPU B durch den Interface-Baustein B
11 Übernahme der übertragenen Nachricht aus dem Arbeitsspeicher zur CPU B

Kommunikationsspeicher = Teil des Arbeitsspeichers

◀ *Bild 9 Grundstruktur eines MMPS auf Multibus-II-Basis*

Bild 10 Nachrichtenaustausch zwischen zwei CPU

Mehrprozessorsysteme müssen die Signalisierung von Interrupts zwischen den Prozessoren ermöglichen. Da die Verwendung von Interruptleitungen wegen des notwendigen Leitungsaufwands von n x (n−1) Leitungen unzweckmäßig ist, überträgt die Multibus-II-Architektur in Anlehnung an den später beschriebenen Kommunikationsmechanismus auf der Basis eines standardisierten Nachrichtenprotokolls Interrupts in Nachrichtenform. Auf diese Art und Weise stehen 256 virtuelle Interrupteingänge und -ausgänge je Steckeinheit zur Verfügung.

3.2.1.4. Adreßräume

Zu Beginn des Buszyklus wird der Adreßraum durch die Kommando- bzw. Statusleitungen SC 4, SC 5 als Basis für die Interpretation der Adresse definiert:
– Speicheradreßraum mit 32-Bit-Adresse, Adressierung von 4 GByte Arbeitsspeicher
– E/A-Adreßraum mit 16-Bit-Adresse, Adressierung von 64 K E/A-Ports.
Diese beiden Adreßräume entsprechen denen herkömmlicher Bussysteme.
– Interconnect-Raum für Installation und Wartung mit 16-Bit-Adresse, davon: 5 Bit für örtliche Adressierung, d. h. Lage der einzelnen Steckeinheit im System, 11 Bit für Adressierung der Register je Steckeinheit mit Herstellungs- und Größenangaben, Initialisierungs- und Wartungsinformation. Dieser Raum dient zur Konfigurierung des Systems durch das Betriebssystem ohne manuelle Eingriffe.
– Message-Adreßraum mit 8-Bit-Adresse: Für logische Adressierung der Steckeinheit zwecks Übertragung von Nachrichten variabler Länge (Bündel-Transfer) ohne Kenntnis der Speicher- oder Portadressen auf der Steckeinheit: Nachricht als standardisiertes Protokoll zur Modulkommunikation mit Hardwareunterstützung für Informationsaustausch

zwischen CPUs, Interruptsignalisierung und Software-Reset.

3.2.2. Lokalbus (iLBX II)

Zur Erhöhung der Leistungsfähigkeit wird die Zugriffszeit durch Verwendung des Lokalbusses verringert, wodurch der Systembus entlastet wird, der bei alleinigem Vorhandensein nur ein MMPS mit maximal drei Prozessoren ermöglicht. Der Lokalbus ist Voraussetzung für die Zuordnung individueller Speicher. Diese unmittelbare Zuordnung ermöglicht den sofortigen Zugriff über den Bus, verwendet leistungsschwächere (damit schnellere) Bustreiber und vereinfacht (oder erspart) die Arbitrierung. Damit ist eine Leistungssteigerung des einzelnen Prozessors und des MMPS möglich.
An den iLBX-II-Bus sind maximal 6 Steckeinheiten anschaltbar, davon können zwei Busmaster sein. Der Arbitriermechanismus ist sehr einfach: Ein Primär-Master hat dauernde Zugriffsmöglichkeit zum Bus, ein Sekundär-Master muß bei Notwendigkeit eine Busanforderung mittels eines Request/Acknowledge-Protokolls an den Primärmaster stellen. Dieser Mechanismus entspricht der Kommunikation zwischen CPU und DMA-FE. Der Primärmaster hat außerdem Zugang zum parallelen Systembus, der Sekundärmaster und die Slaves können je nach Bedarf einen derartigen Zugang erhalten. Das System kann mehrere Lokalbusse enthalten.
Der iLBX-II-Bus hat 26 Adreßleitungen für 64 MByte lokalen Speicher und 32 Datenleitungen. Der dadurch mögliche Verzicht auf Multiplexbetrieb erlaubt eine Überlappung von Daten- und Adreßphasen (pipelining). Der iLBX-II-Bus wird synchron zum Primärmaster betrieben, was eine optimale Anpassung der CPU an den Speicher ermöglicht. Bei einer maximalen Taktfrequenz von 12 MHz ergibt sich eine Transferrate von 48 MByte/s, wobei auch Blocktransfer möglich ist.

3.2.3. Serieller Systembus (iSSB)

Durch den seriellen Systembus sind Verbindungen zwischen Steckeinheiten über eine Zweidrahtleitung möglich. Daraus resultieren für bestimmte Einsatzfälle
– die Senkung der Kosten für den Verbindungsweg (Ersatz von etwa 60 Leitungen des parallelen Systembusses durch zwei Leitungen) und
– die Möglichkeit der Verbindung zwischen örtlich weit auseinanderliegenden Steckeinheiten aus verschiedenen Multibusbereichen und der Zuordnung zu verschiedenen Rückverdrahtungsplatten.

Eine sinnvolle Anwendung besteht im zusätzlichen Betrieb zum parallelen System als Diagnose- und/oder Ersatzkanal.
Der serielle Systembus dient der Übertragung von Nachrichten unterschiedlichen Typs und variabler Länge, die durch einen 16-Bit-CRC geschützt wird. Die Adressierung der Teilnehmer erfolgt mit 8 Bit. Softwaremäßig besteht kein Unterschied zwischen der Informationsübertragung über seriellen oder parallelen Systembus; die Erstellung der Software ist also unabhängig von der Implementierung der Verbindungsstruktur.
Über den Bus lassen sich bis zu 32 Knoten verbinden (Kabellänge bis zu 10 m). Als Übertragungsverfahren wird ein CSMA/CD-Verfahren[1] ohne Arbitriersignale verwendet.
Dieses Verfahren hat folgende Eigenschaften:
– Alle FE sind gleichberechtigt an den Bus geschaltet und registrieren seinen Zustand.[2]
– Ist der Bus frei, kann bei Bedarf das Senden einer Nachricht durch eine FE sofort durchgeführt werden.
– Ist der Bus belegt, muß die FE mit Übertragungsanforderung auf dessen Freiwerden warten.

Kollisionen treten auf, wenn mehrere Sender gleichzeitig senden. Das kann auftreten
– infolge Registrierens des Freiseins des Verbindungsweges
– infolge Feststellens des Endes einer Übertragung.

Das Auflösen von Kollisionen erfolgt mittels Zeitscheibenverfahren, das jeder FE am Bus innerhalb einer maximalen Zeit das Aussenden ihrer Nachricht ermöglicht (deterministisch, TDMA-Verfahren[3]).
Die physikalische Signalisierung verwendet die beiden Leitungen SDA und SDB mit folgender Codierung:

SDA	SDB	Information
0	0	Kollision
0	1	logische 0
1	0	logische 1
1	1	Leitung frei.

3.2.4. Informationsaustausch mit (standardisiertem) Nachrichtenprotokoll

Die Informationsübertragung über den parallelen und seriellen Systembus wird auf der Basis eines Kommunikationsprotokolls als Teil der Busspezifikation mit Hardwareunterstützung nach dem Prinzip von Bild 6 und 7 durchgeführt. Die Anordnung und der Ablauf für Senden und Empfangen der Nachricht sind im Bild 10 dargestellt.
Die unsolicited[4]- und solicited[5]-Übertragung charakterisiert einen Ein- und Mehr-Block-Informationsaustausch. Das Nachrichtenpaket hat folgendes Aussehen:

Zieladresse 8 Bit	Quelladresse 8 Bit	Kommando	Länge	Nachricht	CRC

Längen- und Nachrichtenfeld sind nur in bestimmten Kommandos vorhanden, das CRC-Feld ist nur beim seriellen Sytembus erforderlich.
Die Abläufe nach Bild 10 zeigen,
– daß die Kenntnis der Arbeitsspeicheradressen der CPU B für die CPU A nicht erforderlich ist und
– daß die CPU A und die CPU B nicht direkt am Transfer beteiligt sind.
Es handelt sich also um einen betriebssystemneutralen und schnellen Informationsaustausch-Mechanismus.
Wichtige Kommandos sind:
– Reset Modul: softwaremäßiges Rücksetzen eines Moduls
– Interrupt: virtuelle Interruptübermittlung in Form eines Nachrichtenpakets
– Unsolicited-Nachricht: Übertragungsprotokoll ohne Kommunikationsspeicherzuweisung beim Empfänger für kurze Nachrichten fester Länge
– Solicited-Nachricht: Übertragungsprotokoll mit Kommunikationsspeicherzuweisung beim Empfänger für lange Nachrichten variabler Länge

[1] carrier sense multiple access – stochastische Buszugriffsmethode mit dezentraler Steuerung, Zugriff mit teilweiser Kooperation, Verhindern der Übertragung durch einen Sender bei bereits laufender Übertragung
[2] broadcast-Verhalten: ein Kanal, bei dem alle FE in der Lage sind mitzuhören, um festzustellen, ob eine Übertragung stattfindet, und um zu erkennen, ob eine Nachricht für die eigene oder eine andere FE bestimmt ist
[3] time division multiple access scheme – Zugriff mit vollständiger Kooperation
[4] unmittelbare Übertragung vom Sender zum Empfänger
[5] mittelbare Übertragung nach Aufforderung durch den Sender und anschließender Anforderung durch den Empfänger

4. Vergleich zwischen Multibus I und II

Für die Anwendung einer Bus-Architektur mit verschiedenen Teilbussen gelten folgende Aspekte:
1. Unter Beibehaltung der Bandbreite des allgemeinen Falls wird eine virtuelle Bandbreite für Prozessor-Kommunikation und Datenaustausch erzeugt.
2. Spezielle Busse können ihre Funktionen besser erfüllen als der allgemeine Bus.
3. Die Funktionen können parallel über verschiedene Busse abgearbeitet werden.
4. Nutzer können auf Grund vorgegebener Systemeigenschaften die dem Verwendungszweck entsprechenden Busse auswählen und damit minimale Kosten erzielen.
Der Übergang vom Multibus I zum Multibus II ist zweckmäßig
– bei Notwendigkeit der Erhöhung der Verarbeitungsleistung
– bei Forderungen bezüglich der Verbesserung der Multiprozessorfähigkeit
– bei Notwendigkeit der Anwendung von 32-Bit-Mikroprozessoren
– bei Forderungen bezüglich der Systemzuverlässigkeit.
Als Optimierungsprinzipien für Teilbusse der Multibus-II-Architektur gelten folgende Gesichtspunkte:
– iPSB Optimierung der Prozessor-Kommunikation und des damit verbundenen Datenaustausches
– iLBX II Optimierung bezüglich maximaler Verarbeitungsgeschwindigkeit
– iSSB Optimierung bezüglich minimaler Kosten für Prozessor-Kommunikation.
Vorteile der Multibus-II-Architektur gegenüber der Multibus-I-Architektur:
1. Höhere Transferrate über den parallelen Systembus:
20 bzw. 40 MByte im Einfach- bzw. Mehrfach-(Block-)Zyklus gegenüber 5 MByte (Bytebetrieb) oder 10 MByte (Wortbetrieb)
2. Möglichkeit der Bildung von Prozessorbereichen durch Verwendung des seriellen Systembusses (Entfernung bis zu 10 m)
3. Vergrößerung des Speicherraums von 16 MByte auf 4 GByte durch 32-Bit-Adresse
4. Erhöhung der Übertragungssicherheit durch Verwendung je eines Paritätsbits für 8 Leitungen (Adreß-, Daten-, Kommando- und Steuerleitungen)
5. Möglichkeit der Zusammenarbeit von 8-, 16- und 32-Bit-Rechnern und Durchführung von 8-, 16-, 24- und 32-Bit-Übertragungen gegenüber 8- und 16-Bit-Übertragungen
6. Verwendung eines Übertragungsprotokolls ohne Interrupt zur Verminderung der Störanfälligkeit und des Aufwandes sowie zur Erhöhung der Übertragungsgeschwindigkeit ergibt Unabhängigkeit von Sender/Empfänger und Möglichkeit des Blocktransfers.
7. Anwendung eines virtuellen Interruptkonzepts als Ersatz für ein solches mit Steuerlei-

tungen zum Zweck der Erhöhung der Anzahl der Interrupteingänge und -ausgänge und Verminderung des Aufwandes.

8. Verwendung eines Lokalbusses (iLBX II) mit getrenntem Adreß- und Datenbus und der Möglichkeit einer überlappten Arbeitsweise (pipelining) zur Erhöhung der Verarbeitungsleistung.

9. Erhöhung der Speicherkapazität des Lokalspeichers auf 64 MByte durch Verwendung von 26 Adreßleitungen.

10. Verwendung eines Lokalspeichers mit Zugriffsmöglichkeit durch Prozessor der eigenen oder anderer Lokalbusbereiche mit Erhöhung des Freiheitsgrades der Speicherzuordnung (Unterbringung der Daten im Bereich mit maximaler Nutzung, wodurch sich eine Entlastung des Systembusses ergibt, der in diesem Fall nur bei Nutzung der Daten durch Prozessoren anderer Lokalbusbereiche in Anspruch genommen wird).

Bei Multibus-II-Architekturen können von Prozessoren unterschiedlicher Lokalbus- und Systembus-Bereiche genutzte Daten im Lokal- oder Globalspeicher untergebracht werden, während Multibus-I-Architektur die Speicherung derartiger Daten im Globalspeicher notwendig macht.

Literatur

/1/ Geyer, J.: Multibus II: 32-bit-Bus für leistungsfähige offene Systeme. Elektronik 26/30. 12. 1983 S. 32–37 und 1/13. 1. 1984 S. 2–5

/2/ Jacks, E.: Multibus Link preserves hardware and software investment. update 8 (1984) 6, S. 297–300

/3/ Müller, K. D.: Comparison and Status of 32 Bit Backplane Bus Architectures. IEEE Nuclear Science Symp. 1984 Orlando/Florida

EPROM-Programmiergerät EPG01

Ekkehard Fischer, Andreas Edelmann
VEB Erdöl-Erdgas Grimmen

Mit dem beschriebenen Programmiergerät ist es möglich, EPROMs von 2 bis 32 KByte Speichervermögen zu programmieren (Tafel 1). Mit einer unkomplizierten Hard- und Softwaremodifizierung lassen sich auch EPROMs bis 64 KByte programmieren, was aber nicht Anliegen dieses Beitrages sein soll. Das Gerät läßt sich an einen PC-1715-Bus anschließen. Auf die Programmierung von 1-K-EPROMs wurde keine Rücksicht genommen, da sie weitgehend veraltet sind und unkomfortable Anschlußbedingungen besitzen. Die erforderliche Software ist in TURBO-PASCAL erarbeitet worden und bietet außer der Software für die Programmierung noch weitere Bedienerserviceroutinen. Eine Erweiterungsmöglichkeit des PC 1715 stellt die separat einsetzbare parallele Schnittstelle dar. Die Nachnutzung des gesamten Systems bzw. einzelner Komponenten ist über das BfN des VEB Erdöl-Erdgas Grimmen möglich.

Tafel 1 Programmierbare EPROM-Typen mit der Angabe der Programmierspannung und dem Speichervolumen

EPROM-Typ	Speicher-volumen	Programmier-spannung
2716	2	25 V
2732	4	25 V
2732 A	4	21 V
2764	8	21 V
2764 A	8	12,5 V
27128	16	21 V
27128 A	16	12,5 V
27256	32	12,5 V

Tafel 2 I/O-Portadressen des PC 1715 für die parallele Schnittstelle

I/O-Portadresse	Funktion
80H	PIO 1A Daten
81H	PIO 1B Daten
82H	PIO 1A Contr.
83H	PIO 1B Contr.
84H	PIO 2A Daten
85H	PIO 2B Daten
86H	PIO 2A Contr.
87H	PIO 2B Contr.
88H	CTC Kan. 0
89H	CTC Kan. 1
8AH	CTC Kan. 2
8BH	CTC Kan. 3

Allgemeiner Aufbau

Das EPROM-Programmiergerät EPG01 besteht im wesentlichen aus folgenden drei Baugruppen (Bild 1):

– parallele Schnittstelle
– Programmiereinheit mit Programmierspannungsversorgung
– EPROM-Steckeinheit.

Die Untergliederung des Gesamtgerätes wurde deshalb vorgenommen, da die parallele Schnittstelle separat nutzbar sein sollte.

Beschreibung der Funktionsgruppen

Parallele Schnittstelle

In ihrer Grundkonfiguration ist die parallele Schnittstelle mit zwei PIOs U 855 C ausgestattet. Auf der Leiterkarte der parallelen Schnittstelle besteht die Möglichkeit, einen CTC U 858 C einzusetzen. Damit ist man in der Lage, weitere Anwendungsfälle für diese Baugruppe zu finden.

Dipl.-Ing. Ekkehard Fischer (31) studierte von 1974 bis 1979 am Aserbaidshanischen Institut für Erdöl und Chemie (UdSSR), Sektion Bohrtechnik. Von 1983 bis 1985 absolvierte er ein postgraduales Studium an der Technischen Hochschule Ilmenau, Sektion Technische und Biomedizinische Kybernetik. Seit 1986 arbeitet er als Forschungsingenieur in der Abteilung Mikroelektronik des VEB Erdöl-Erdgas Grimmen und beschäftigt sich mit dem Einsatz der Mikroelektronik auf Bohranlagen.
Dipl.-Ing. Andreas Edelmann (29) studierte von 1981 bis 1986 an der Technischen Hochschule Ilmenau, Sektion Technische und Biomedizinische Kybernetik. Seit 1986 arbeitet er als Leiter der Abteilung Mikroelektronik des VEB Erdöl-Erdgas Grimmen. Seine derzeitige Forschungsarbeit befaßt sich mit der Prozeßdatenerfassung und -auswertung im Bohrprozeß.

Alle Elemente der vollständigen Schnittstelle sind interruptfähig. Durch eine Umgehungslogik wurde die Interrupteinschwingzeit minimal gehalten /2/. Der beim PC 1715 freie I/O-Port-Adreßbereich von 80H bis 8BH fand bei Belegung durch die parallele Schnittstelle Verwendung. Eine entsprechende Untergliederung der I/O-Portadressen ist in Tafel 2 zu finden. Die Zuordnung der PIO-Ports zu den von der Programmiereinheit benötigten Signalen ist in Tafel 3 dargestellt.

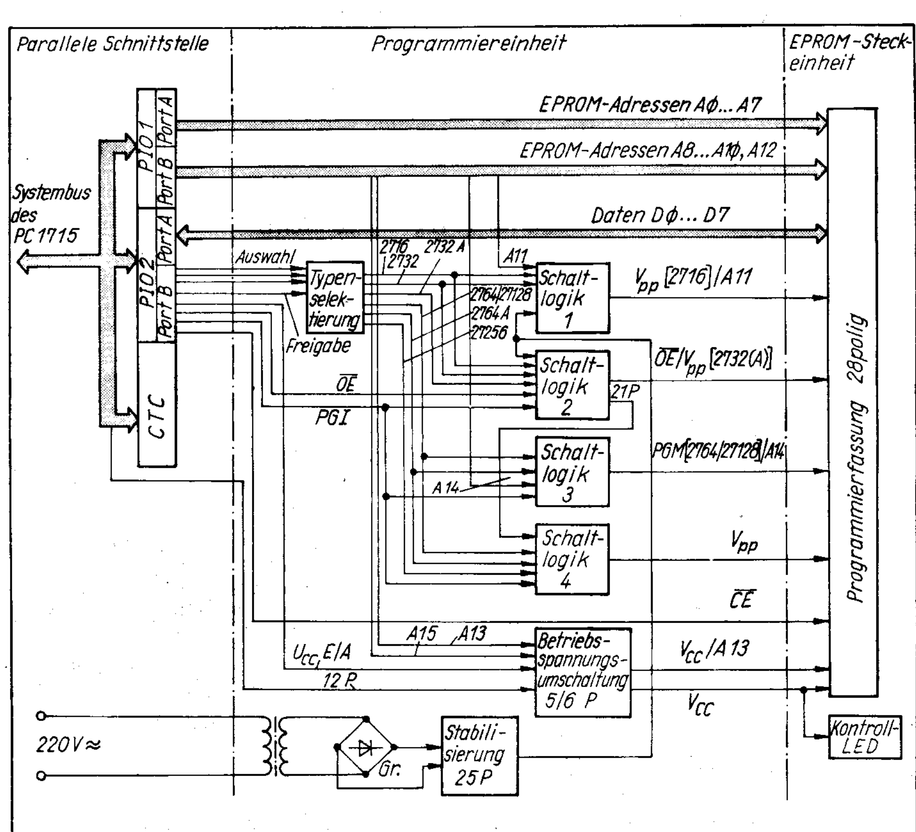


Bild 1 Blockschaltbild

Tafel 3 Zuordnung der PIO-Ports zu den Signalen der Programmiereinheit

PIO-Port	Bit	Funktion
A1	0...7	EPROM-Adreßbit 0...7
B1	0...6	EPROM-Adreßbit 8...14
	7	Umschaltung der Betriebsspannung 5 V bzw. 6 V
A2	0...7	EPROM-Datenbit 0...7
B2	0...2	Typenauswahl
		Bit 0 1 2 Typ
		0 0 0 2716
		0 0 1 2732
		0 1 0 2732 A
		0 1 1 2764/27128
		1 0 0 2764 A/27128 A
		1 0 1 27256
	3	Dekoderfreigabe
	4	Betriebsspannungsfreigabe
	5	OE
	6	/PGI
	7	/CE

Programmiereinheit

Auf der Leiterkarte der Programmiereinheit sind folgende Funktionsgruppen implementiert:

- Typenselektierung
- Schaltlogik 1 bis 4
- Programmierspannungserzeugung
- Betriebsspannungsumschaltung 5/6 V.

Von der Programmiereinheit werden alle die Signale und Spannungen erzeugt, die benötigt werden, um EPROMs der in Tafel 1 genannten Typen zu programmieren, zu verifizieren und auszulesen. Dadurch, daß die Programmiereinheit auch eine Programmierspannung von 12,5 V zur Verfügung stellt, besteht die Möglichkeit, CMOS-EPROMs zu programmieren. Die Steuerung der Programmiereinheit wird durch ein entsprechend angepaßtes Softwarepaket des PC 1715 über die parallele Schnittstelle realisiert. Damit wird vermieden, daß durch evtl. Fehlschaltungen (z. B. mit DIL-Schaltern) eine Zerstörung der EPROMs stattfinden kann. Anhand der Typenselektierung (1-aus-8-Decoder 8205) wird die Schaltlogik 1 bis 4 aktiviert und dem ausgewählten EPROM-Typ der jeweilige Pegel des Programmiersignals geliefert. Weiterhin sorgen die Schaltlogiken für ein typengerechtes Signalspiel (Bild 2) während des Programmablaufs. Die genannten unterschiedlichen Programmierspannungen werden aus einer 25-V-Spannungsquelle gewonnen und je nach Programmiervorschrift und Selektierung zugeschaltet. Um die Programmierzeit zu verkürzen (besonders bei Typen über 4K) und außerdem einen dementsprechenden Programmieralgorithmus (INTEL) zu verwenden, sorgt eine schaltbare Spannungsquelle für zwei verschiedene Betriebsspannungen (5 bzw. 6 V) /1/.

EPROM-Steckeinheit

Außer der 28poligen Programmierfassung ist noch eine LED auf der EPROM-Steckeinheit implementiert. Über diese wird dem Nutzer der Zugriff des Programmiergerätes auf dem EPROM angezeigt. Um Zerstörungen der EPROMs vorzubeugen, ist anzuraten, einen EPROM-Wechsel beim Leuchten der LED zu unterlassen.

Software

Das Softwarepaket zum Betrieb des EPROM-Programmiergerätes besteht aus einzelnen TURBO-PASCAL-Prozeduren, welche über einen Bedienerdialog im Hauptprogramm aufgerufen werden. Mit dem Programmruf „EPROM" wird das Programm aus SCP o. ä. gestartet und belegt den Speicherbereich von 100H bis C342H des PC 1715. Nach dem Start meldet sich das Programm mit dem in Bild 3 abgebildeten Hauptmenü. Aus diesem wird über Tastendruck ohne Quittierung durch ET in die Prozeduren verzweigt. Es ist dabei einerlei, ob Groß- oder Kleinbuchstaben verwendet werden.

Die Arbeit mit dem gesamten Programm ist unkompliziert. Durch die Dialogführung des Bedieners werden Bedienerfehler weitestgehend ausgeschlossen bzw. signalisiert. Im großen und ganzen untergliedert sich das Menü in drei Hauptgruppen. Im folgenden werden die Arbeitsmöglichkeiten kurz beschrieben.

EPROM-Bearbeitung

Der erste Arbeitsschritt bei der EPROM-Bearbeitung sollte die Typenauswahl sein, da sich alle anderen Routinen darauf beziehen. Vor der Programmierung oder dem Auslesen der EPROMs werden dem Bediener Anweisungen zum Wechseln des EPROMs gegeben. Bei der Programmierung wird der o. g. Algorithmus angewandt. Ist ein Speicherplatz ordnungsgemäß programmiert, wird das durch die Ausgabe der Speicherplatzadresse signalisiert. Besteht keine Möglichkeit, den EPROM auf dem selektierten Speicherplatz zu programmieren oder ist die Steckfassung leer, so wird eine entsprechende verbale Aussage mit Adreßangabe auf dem Bildschirm ausgegeben.

Pufferarbeit

Über das Programm zum Editieren des Puffers können Veränderungen bzw. Neueintragungen cursorgeführt vorgenommen werden. Dazu wird ein Ausschnitt des Puffers als Dump auf dem Bildschirm abgebildet. Weitere Routinen bieten die Möglichkeit, Pufferinhalte zu verschieben, den Puffer mit einem entsprechenden Kennbyte oder einem ASCII-Zeichen zu füllen.

Dateiarbeit

Der zur Programmierung der EPROMs eingerichtete Datenpuffer kann mit Daten aus .HEX-, .TXT- oder .COM-Dateien von der Diskette geladen bzw. es kann der Inhalt des Puffers auf Diskette ausgelagert werden. Beim Laden des Puffers werden die Blocknummern der geladenen Dateien angezeigt. Anhand weiterer Routinen können COM-Dateien cursorgeführt gestartet und das Diskettenverzeichnis angezeigt werden.

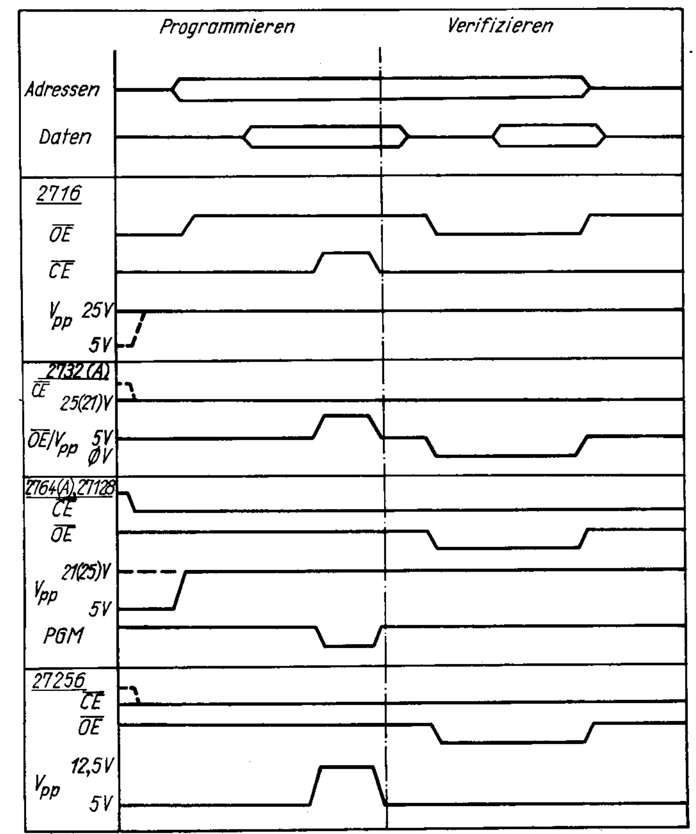


Bild 2 Signalspiel beim Programmieren und Verifizieren der EPROM-Typen 2716, 2732 (A), 2764 (A), 27128, 27256

```
EPROM-Type 2732 A wurde ausgewaehlt

    Dialog zur Bedienung des EPROM - Programmiergeraetes EPR01

    folgende Funktionen koennen ausgewaehlt werden :

                                     Taste                                   Taste

Typenauswahl                           T      EPROM aus Puffer programmieren   P
EPROM-Leerpruefung                     B      EPROM in Puffer laden            L

Pufferdump                             D      Puffer mit Konst. fuellen        F
Puffer editieren                       M      Pufferbereich verschieben        S

Puffer in HEX-Datei schreiben          W      Puffer in COM-Datei schreiben    U
Puffer aus HEX-Datei laden             R      Puffer aus COM-Datei laden       G

Puffer in Textdatei schreiben          A      Directory anzeigen               H
Puffer aus Textdatei laden             X      Ende                             E
Start einer COM-Datei                  I
                                              Bitte auswaehlen :
```

Bild 3 Bildschirmausgabe des Hauptmenüs

Literatur

/1/ Behnke, H.: EPROMs sind auch nur Speicher. c't 1985, Heft 2, S. 88–92

/2/ Kieser, H.; Meder, M.: Mikroprozessortechnik, 4. Aufl. Berlin: VEB Verlag Technik 1986

☒ KONTAKT ☏

VEB Erdöl-Erdgas Grimmen, Abt. Mikroelektronik, Stoltenhäger Chaussee, Grimmen, 2320; Tel. 4 30 (Anfragen zur Nachnutzung bitte an Abt. BfN, Kolln. Fitsch)

Ein Computerclub stellt sich vor

Computertagung in Frankfurt (Oder)

Im November vergangenen Jahres veranstaltete der Computerclub Frankfurt (Oder) die 4. Computertagung. Etwa 400 Teilnehmer tagten am 4. und 5. November in der Stadthalle von Frankfurt (Oder). Wegen der großen Resonanz und der Vielzahl an Themen fand die Veranstaltung erstmals an 2 Tagen statt. Organisiert vom Computerclub in Frankfurt (Oder), der im Bezirk die Funktion eines Leitclubs inne hat, stand die 4. Tagung unter Leitung des Schwedter Kreisclubs. Gemischt wie der Teilnehmerkreis war auch das Spektrum der Vorträge. Angesprochen wurden sowohl Amateure als auch Fachleute, die sich beruflich mit der Thematik beschäftigen.



1

Zum Vortragsprogramm gehörten u. a. Beiträge über Bauelemente, Eigenbaucomputer, -peripherie und die Programmiersprache FORTH ebenso wie Erfahrungen über Probleme der Clubarbeit. Zwei Konsultationspunkte – zu FORTH und zum KC 85/2(/3) – ordneten sich thematisch in das Vortragsprogramm ein. An den K-Punkten war eine Übersicht zu Hard- und Softwarelösungen erhältlich, die auch eine Literaturzusammenstellung zu FORTH enthielt. Leider war in der recht umfangreichen Übersicht kein Hinweis auf die Veröffentlichungen zu FORTH in MP 6/87 zu finden.

Der Computerclub Strausberg stellte gleich drei interessante Lösungen vor. Die 256-K-dRAM-Erweiterung für KC 85/2(/3) – ausführlich vorgestellt in MP 12/87, S. 373 –, ein EPROM-Programmiergerät für EPROM-Module des Herstellers und ein 32-K-EPROM-Modul sollen nachfolgend kurz beschrieben werden (Bild 1).

Das Modulkonzept des Herstellers, insbesondere bei ROM- und PROM-Modulen sieht vor, daß EPROMs perspektivisch auch auf der Leiterkarte des jeweiligen Moduls zu programmieren sind. Auf den vorhandenen Modulen sind deshalb alle erforderlichen Anschlüsse herausgeführt und Bestandteil des Bussystems der Rechner. Diese Tatsache wird bei der vorgestellten Lösung genutzt. Ausgehend von den Erfahrungen mit dem EPROM-Programmiergerät in /1/ wurde ein Gerät aufgebaut, das zwar nach dem gleichen Prinzip arbeitet, aber wesentlich geringer im Aufwand bei der Realisierung ist. Weitere Vorteile sind darin zu sehen, daß die EPROMs nicht gezogen und wieder gesteckt werden müssen, sondern im Modul verbleiben können, sofern sie gelöscht sind bzw. die Möglichkeit besteht, sie im Modul zu löschen, was der Hersteller durch die Fenster im M025 offensichtlich auch vorgesehen hat. Weiterhin wird keine teure und leider immer noch sehr schwer beschaffbare Schwenkhebelfassung benötigt. Die Realisierung erfolgte auf einem Adaptermodul, wo eine Lochrasterkarte mit der Schaltung (mit einem Schaltkreis DL123D) montiert wurde. Die Programmierspannungserzeugung erfolgt extern über ein Steckernetzteil. Der 32K-EPROM-Modul wurde auf Basis des Modules M006 (BASIC) realisiert. Von diesen 32K liegen jeweils 8K im Adreßraum des Prozessors. Das ist mit allen vier 8K-Bereichen möglich. Eine weitere Besonderheit des Moduls besteht darin, daß die vier beschriebenen Bereiche auf je zwei Basisadressen verfügbar sind, und zwar auf den Basisadressen 4000H und C000H, um spezielle Betriebssystemfunktionen wie „DISPLAY" nutzen zu können. Die Verwaltung des Moduls (Einschalten, Festlegen der Basisadresse, Auswählen des jeweiligen EPROM-Bereiches usw.) geschieht ausschließlich softwaremäßig, d. h., es sind weder DIL-Schalter noch Wickelstifte o. ä. zur Adreßdekodierung erforderlich. Zur praktischen Realisierung des Moduls sei noch erwähnt, daß die beiden ROM-Schaltkreise des Originalmoduls durch zwei Schaltkreise 27128 (EPROM 16K×8) zu ersetzen sind. Weitere zusätzliche Schaltkreise sind nicht erforderlich. Es wurden lediglich an der Beschaltung einiger Bauelemente Änderungen vorgenommen.

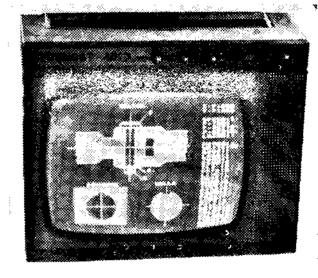
Ebenfalls im K-Punkt wurde ein Grafikprogramm von M. Feindt zum Erstellen und Bearbeiten von Konstruktionszeichnungen auf dem Bildschirm für KC 85/3 demonstriert (Bild 2).

Das Programnm ist in Maschinencode geschrieben und benötigt zum Grundgerät KC 85/3 V.24- und 16-KRAM-Modul sowie Monitor und Drucker. Gezeichnet wird auf einer Fläche mit 255 × 255 Bildpunkten, wobei für die Nutzerführung auf der rechten Bildschirmseite ständig das Menü mit allen Funktionen und deren Erreichbarkeit eingeblendet ist. Der Cursorpunkt kann mit den Cursorsteuertasten in 8 Grundrichtungen und 10 einstellbaren Geschwindigkeitsstufen bewegt werden. 8 Befehlsgruppen sorgen für den Komfort des Programms. So gibt es Befehle für Stricharten und -stärken, neben der nichtunterbrochenen Linie sind Strich-Strich-Linien und Strich-Punkt-Linien möglich, Befehle zum Zeichnen ebener Figuren: Kreis, Quadrat, Rechtecke, Gerade mit beliebigem Anstieg und eine Fläche mit einem Durchmesser von 3 Punkten (z. B. für Lötaugen) sowie sogenannte Flächenbefehle. Mit ihnen können geschlossene Figuren ausgefüllt oder schraffiert werden. Im entsprechenden Modus stehen dem Benutzer in der 2. Ebene der Tastatur 26 Quasigrafikzeichen zur Verfügung. Angeboten werden spezielle Elektronik- und Maschinenbausymbole sowie allgemeine Symbole. Die erstellten Grafiken können auf dem Bildschirm verschoben werden, sie lassen sich drehen und vergrößern bzw. verkleinern. Bildschirminhalte können auf Kassette oder Drucker ausgegeben sowie von Kassette eingelesen werden. Software und Hardware werden den Forderungen, die an ein „echtes" CAD-System bestehen, sicher nicht gerecht. Dennoch dürfte das ausgestellte Programm sehr gut für die Aus- und Weiterbildung auf diesem Gebiet geeignet sein.

3

Die NVA Frankfurt (Oder) stellte einen Trainer für die Fernschreibausbildung vor (Bild 3). Das BDT 87-1, es enthält CTC-Baustein, 3 PIO und diverse Koppelelektronik, ist direkt an einen KC 85/3 angeschlossen. Das Gerät ermöglicht die Kontrolle von maximal 24 Fernschreibmaschinen. Verschiedene Betriebsarten unterstützen den Lehrer bei der Ausbildung. So erlaubt die Betriebsart „Vorgabezwang", für die Lernenden eine bestimmte Anzahl von Anschlägen je Minute vorzugeben. Anhand einer Bildschirmgrafik lassen sich die an jedem Platz erreichten Ergebnisse darstellen. Bei der Betriebsart Leistungskontrolle wird der zu schreibende Text vorher in den Computer gespeichert, der nach der Leistungskontrolle durch zeichenweisen Vergleich die Auswertung und Zensierung vornimmt.

Starker Andrang herrschte am Stand, wo der Heimcomputer HCX – Basis MP U880 – vorgestellt wurde (Bild 4). Ein Kollektiv der TU Magdeburg hatte diesen Computer entwickelt. Das Besondere: er verfügt über ein Minimum an Hardware. Aufgrund des besseren Angebots an ROM-Bauelementen können mehr Funktionen über Software realisiert werden, dadurch sind Abstriche an der Hardware möglich. Das Zusammenspiel mit den Peripherieschaltkreisen wird mittels PIO, CTC und Buspuffer realisiert. Relativ beliebige Bildschirmformate lassen sich leicht nachbilden (grafisch oder nichtgrafisch), z. B. von allen KCs. Auch das BASIC-Betriebssystem der



2

4

KC ist leicht implementierbar. Als zweckmäßig hat sich ein ZX-Spectrum-kompatibles Betriebssystem erwiesen. Das Grundgerät ermöglicht flexible Bestückungsvarianten mit RAM- und ROM-Schaltkreisen. Der HCX läßt sich über zwei verschiedene Erweiterungssteckverbinder, davon einer K-1520-kompatibel, leicht erweitern. Die Möglichkeit einer Produktion des HCX in kleiner Stückzahl wird gegenwärtig geprüft. Weiterhin stellte der KDT-Computerclub Schwedt Entwicklungen seiner Mitglieder vor, so z. B. einen Eigenbauplotter.

Auch die Tagungspausen wurden zum intensiven Erfahrungsaustausch genutzt. Fachfragen wurden diskutiert, Adressen getauscht und Erfahrungen in der Klubarbeit weitervermittelt. Dabei kamen auch Probleme zur Sprache, so klagten Mitglieder und

30. ZMMM

Eindrücke

Vom 9. bis 20. November 1987 fand die nunmehr 30. Messe der Meister von morgen statt – eine Jubiläumsmesse, die den über 215000 Besuchern erstmals bot, was von vielen Jugendlichen bisher vermißt wurde: neben der Präsentation von über 2000 Exponaten auch viel Wissenswertes, in unterhaltsamer Form dargeboten. Dazu gehörte beispielsweise der Programmierwettbewerb der FDJ, zu dem die „Junge Welt" aufgerufen hatte und für den im Aktionszentrum Computertechnik die entsprechende Hardware zur Verfügung stand (Bild 1). Auch andere Veranstaltungen hatten die Computertechnik zum Thema: Im Forum 87 stellte sich zum Beispiel die Technische Universität Magdeburg mit verschiedenen CAD/CAM-Demonstrationen vor; der Direktor des Informatikzentrums des Hochschulwesens an der TU Dresden, Prof. Dr. rer. oec. Horst Tzschoppe, erläuterte im Fernsehaktionszentrum nicht nur Aufgaben und Vorhaben seiner Institution, sondern vermittelte unter anderem auch viel Interessantes zu Fragen der Softwareerstellung (Bild 2). Schade nur, daß gerade diese Veranstaltung durch ungenügende äußere Bedingungen nicht ihre volle Wirkung auf das Publikum erreichen konnte – ein Mangel, der bei der Vorbereitung künftiger Veranstaltungen ausgeschlossen werden sollte. Auch beim Angebot von Exponaten ließe sich noch einiges verbessern. Wenn man bedenkt, daß jede nachgenutzte Lösung Einsparung geistigen Potentials bedeutet, ist es nicht vertretbar, daß eine Messe – lt. Fremdwörterbuch Verkaufsveranstaltung – von vielen Ausstellern nicht besser genutzt wird, um über ihr Exponat zu informieren. Mit dem Verweis auf Interessentenlisten zum Anfordern von Informationsmaterial oder auf Kataloge, die von Fachministerien beziehbar sind, ist es nicht getan. Notwendig sind sofort verfügbare Informationsblätter, eine qualifizierte Standbetreuung und nicht zuletzt – seitens der Veranstalter – eine bessere Öffentlichkeitsarbeit. Denn eine Vielzahl von Fachzeitschriften – auch die MP – ist zwar bereit, die Nachnutzung mit geeigneten Veröffentlichungen zu fördern, wird aber nur ungenügend informiert. Es gibt also noch Reserven, um Mehrfachentwicklungen zu reduzieren – beispielsweise bei „rechnergestützten Arbeitsplätzen für ...", die auch auf dieser ZMMM wieder zahlreich angeboten wurden. Gerade im Bereich der Computertechnik ist es wichtig, die Nutzer viel umfassender und schneller über Vorhandenes zu informieren.

Exponate

Beginnen wollen wir unsere Exponate-Schau mit einem gelungenen Modell der Hochschule für industrielle Formgestaltung Halle Burg Giebichenstein. Das *„Geräteträgersystem mit integrierten Hauptfunktionsgruppen für den CAD/CAM-Arbeitsbereich"* (Bild 3) dürfte bei vielen Besuchern starkes Interesse, bei etlichen angesichts derzeitiger Arbeitsplatzgestaltung aber auch Wehmut hervorgerufen haben. Zu hoffen ist, daß solche sowie universell nutzbare Lösungen wegen der rasch wachsenden Zahl von Computeranwendern schneller praxiswirksam werden.

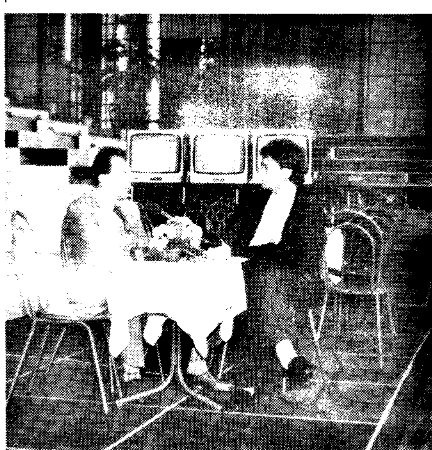
Ein Jugendforscherkollektiv entwickelte für den neuen Robotron-PC, den EC 1834 (Bild 4), den *Farbgrafikadapter*. Er ist kompatibel zum IBM-XT-Standardgrafikadapter CGA, gegenüber diesem jedoch auf zwei Leiterkarten untergebracht. Als Controller wird der U82720 DC03 verwendet. Möglich ist die Darstellung von 16 Farben aus 4096 bzw. von 16 Graustufen auf dem Monochrombildschirm. Im Alphamodus sind 40 × 25 oder 80 × 25 Zeichen darstellbar, im Grafikmodus 320 × 200, 640 × 200, 640 × 400 oder 640 × 480 Punkte. Damit kann sowohl kommerzielle als auch CAD-Grafik realisiert werden. *(VEB Robotron-Buchungsmaschinenwerk Karl-Marx-Stadt, BfN, Annaberger Straße 93, Karl-Marx-Stadt, 9048))*

Das **Intelligente Farbvollgrafikdisplay** ist eine Erweiterung für den PC 1715 zur Darstellung von Farbbildern auf einem handelsüblichen Fernsehbildschirm in 512 × 256 Pixeln, wobei jeder Pixel mit 16 aus 4096 Farben programmierbar ist (Bild 5). In einem Gehäuse befinden sich ein Slave-Rechner, die Grafik und das Netzteil. Der Masterrechner übergibt die Grafikbefehle über die V.24- oder eine nachrüstbare parallele 16-Bit-Schnittstelle an den Slave zur Ausführung. Dies können u. a. sein: interaktive Eingabe durch die Maus (im Bild vorn), Füllen von Flächen mit Farben oder Mustern, Ausgabe von ASCII-Zeichen in beliebiger Größe und kursiv, vom Anwender definierbare Makros. Neben der Grafikerweiterung wird zur Nachnutzung auch die Maus angeboten (Dokumentation). *(IHS Berlin, Prorektorat Naturwissenschaft und Technik, Marktstr. 9, Berlin, 1134)*

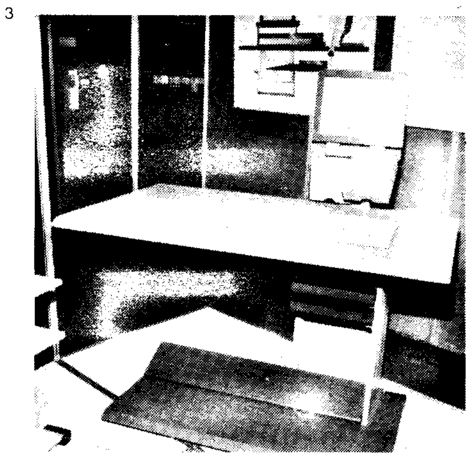
Mit dem **mobilen Datenerfassungsgerät** können bis zu 1024 numerische Daten mit 8 Stellen zeilenweise eingegeben und in einem RAM gespeichert werden (Bild 6). Der eingegebene Wert wird auf einer Flüssigkristallanzeige dargestellt und kann vor dem Abspeichern korrigiert werden. Ein zweites Display zeigt die aktuelle Speicherbelegung. Anschließend können die Daten über eine SIF-1000-Schnittstelle aus dem RAM direkt in einen Mikrorechner übertragen werden. *(VEB Datenverarbeitungszentrum Rostock, Erich-Schlesinger-Str. 37, Rostock, 2500)*



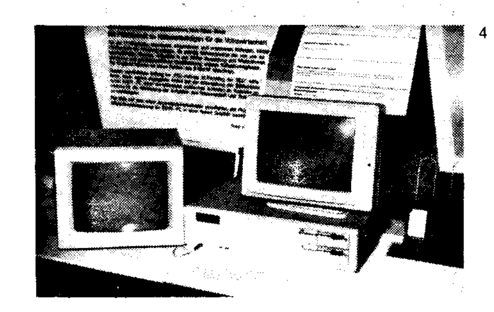
1



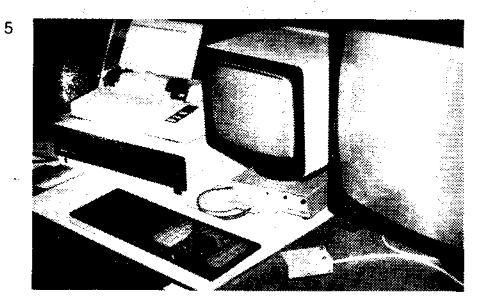
2



3



4



5

Fortsetzung von Seite 90

Leiter von Computerklubs über mangelnde gesellschaftliche Anerkennung ihrer Arbeit. Der gesellschaftliche Nutzen der Klubarbeit besonders auf dem Gebiet der Aus- und Weiterbildung wird von vielen leider noch nicht erkannt. Während die Tagung noch im Gange war, machten die Veranstalter sich bereits Gedanken um die Computertagung 1988. Die 5. soll ganz im Zeichen des KC 85/3-Systems stehen.

Man ist bemüht, diese Tagung in enger Zusammenarbeit mit dem Hersteller durchzuführen. Interessenten, die mit ihren Lösungen zum Gelingen der 5. Tagung beitragen möchten, melden sich bitte beim Computerclub Frankfurt (Oder).

Die Tagung war sehr gut organisiert und hat ihrem Anliegen, ein breites Spektrum an Themen für einen großen Kreis von Interessenten zu bieten, entsprochen.

Kritisch bleibt lediglich anzumerken, daß bei der nächsten derartigen Veranstaltung von Schülern nicht 160,– Mark Tagungsgebühren (für KDT-Mitglieder 60,– Mark) verlangt werden sollten.

I. P., A. B.

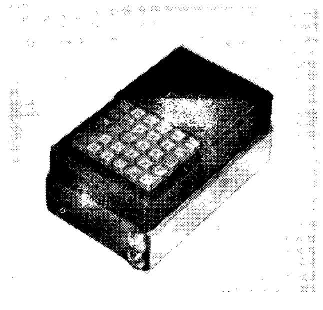
Literatur:

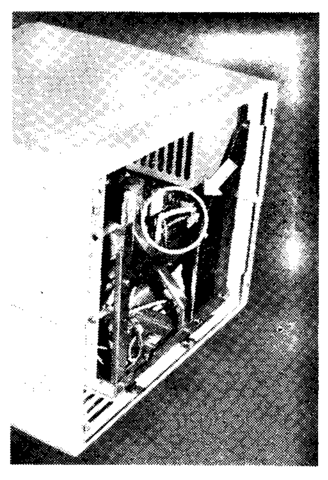
/1/ Völz, H.; Cieri, M.: Einfaches Programmiergerät für KC 85/2 und KC 85/3 Radio, Ferns., Elektron. 36 (1987) 10, S. 662

✉ KONTAKT ☎

Computerclub der KDT Frankfurt (Oder), Ebertusstr. 2, Frankfurt (Oder), 1200

Ein Jugendneuererkollektiv der Druckerei Neues Deutschland entwickelte einen BAS-Modul, um eine **Mehrfachbildschirmanzeige für PC** zu ermöglichen. Verwendet werden Fernsehgeräte, die über einen Videoeingang mit dem PC 1715 (oder A 7100/A 7150) so gekoppelt werden, daß die Sichtbarkeit des vollen Monitorbildes auf den Fernsehbildschirmen gewährleistet ist. Dabei sind Leitungswege bis etwa 100 m ohne Einschränkung der Bildqualität möglich; erprobt wurde bisher der Anschluß von bis zu 20 Fernsehgeräten an einem PC. Zum Leistungsumfang gehört das Umrüsten des PC (Adaptereinbau, siehe Bild 7) und der Fernsehgeräte. (*Druckerei Neues Deutschland, BfN, Franz-Mehring-Platz 1, Berlin 1017*)





In dem **Prüf- und Reparaturplatz für Minifloppylaufwerke** (Bild 8) werden mittels Einchipmikrorechner sowohl die einzelnen Interfacesignale geschaltet als auch Komplexfunktionen realisiert, z. B.: Messung von Umdrehungszeit und evtl. vorhandener Taktfrequenz, Bereitstellung von Schreibdaten, Anzeige aller Steuer- und Statusleitungen, Positionieren des Kopfes in beliebiger Spur, Vorwahl und Prüfen einer Schrittzeit. An dem Arbeitsplatz lassen sich Laufwerke der Typen MFS 1.2 bis MFS 1.6 überprüfen und reparieren. Es ist eine Produktionsübernahme des Testers durch den VEB Robotron-Buchungsmaschinenwerk Karl-Marx-Stadt vorgesehen. (*NVA-Dienststelle, PFN 26212, Strausberg, 1260*)

Bis zu drei Leiterplatten gleichzeitig kann der **Leiterplatten-Bohrautomat LPBM 1.2** (Bild 9) bohren. Dabei bewegt sich der Tisch, von 3 Schrittmotoren getrieben, mit max. 40 mm/s in xy-Richtung. Kernstück der Steuerung ist die ZRE 42521 mit einem K-1520-Bus. (*VE Kombinat Präcitronic Dresden, Stammbetrieb, Fetscherstr. 72, Dresden, 8019*)

In einer Forschungs- und Entwicklungskooperation zwischen dem Informatikzentrum an der TU Dresden und dem VEB Robotron-Projekt Dresden wurde für den Arbeitsplatzcomputer A 7100 (unter dem Betriebssystem SCP 1700) das COBOL-Programmiersystem AC-COBOL geschaffen. Es bietet insbesondere eine gute Unterstützung der Dateiarbeit, wie sie zur Lösung vor allem kommerzieller Probleme und bei der Massendatenverarbeitung notwendig ist. AC-COBOL garantiert volle Sprachkompatibilität zu COBOL-1630; eine Portierung für den A 7150 wird zur Zeit entwickelt. (*Informatikzentrum des Hochschulwesens an der TU Dresden, Mommsenstr. 13, Dresden, 8027*)

Der auf zwei Leiterkarten untergebrachte **Schnittstellenkonverter** (Bild 10) beinhaltet eine Anpaßschaltung zur Übertragung von BCD-Meßwerten in einen Computer über eine V.24-Schnittstelle (ASCII-Code). Damit entfällt die zeitraubende und fehleranfällige manuelle Eingabe. Der Konverter ist insbesondere für den Anschluß von ABBE-Längenmeßgeräten geeignet. (*VEB Barkas-Werke Karl-Marx-Stadt, BfN, Karl-Winzer-Straße, Karl-Marx-Stadt, 9048*)

Zum Betreiben eines Fernschreibers anstelle eines Druckers am KC 85/1 bzw. KC 87 wurde ein **Fernschreibanschluß** entwickelt (Bild 11, vorn rechts), der mittels Hard- und Software die Synchronisation des Rechners mit dem Fernschreiber und die Umcodierung des ASCII- in den Fernschreibcode übernimmt. (*NVA, PF 15513/V, Neubrandenburg, 2040*)

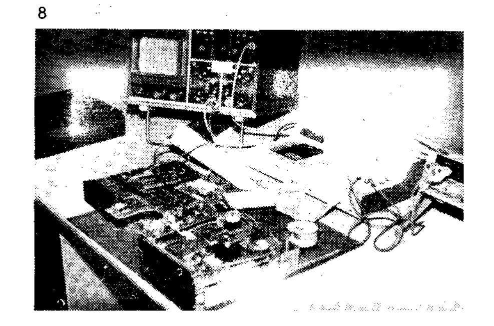
Ein **Teleprinter** genanntes Zusatzgerät zum Anschluß eines Fernschreibers an den KC 85/2 bzw. /3 stellte die *Betriebsschule der Deutschen Post Magdeburg* vor. Auch hier kann ohne Eingriff in den KC der Fernschreiber als Drucker genutzt werden (Bild 12).

Mit dem in der BBS „Conrad Blenkle" hergestellten **Lichtstift** für KC 85/2 bzw. /3 können auf dem Bildschirm Bildpunkte gesetzt und gelöscht werden; d. h., es ist z. B. das „Zeichnen auf dem Bildschirm" möglich. Die Elektronik der Bildpunktadressierung befindet sich auf einer Leiterkarte (siehe Bild 13, vorn rechts auf dem KC-Basisgerät) und besteht ausschließlich aus DDR-Bauelementen. (*VEB Schiffselektronik „Johannes Warnke", PSF 85, Rostock, 2510*)

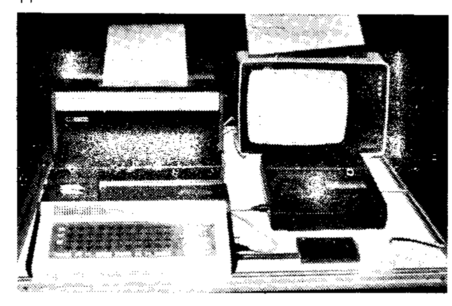
Als Gemeinschaftsarbeit verschiedener Einrichtungen entstand die Lösung der **Vernetzung eines Computerkabinetts** (Bild 14), mit der bereits bis zu 12 KC 85 verschiedener Versionen (da der Kassettenausgang genutzt wird) verbunden werden konnten. Über ein Bussystem sind alle Rechner gleichberechtigt angeschlossen, es lassen sich jedoch auch Master-Slave-Systeme aufbauen. (*Betriebsschule „Franz Günther" des VEB Werk für technisches Glas Ilmenau, Am Ehrenberg, Ilmenau, 6300*)

Der Mikrorechnerbausatz Z 1013 läßt sich nun aufrüsten, und zwar mit den **Erweiterungsbaugruppen** (Bild 15).

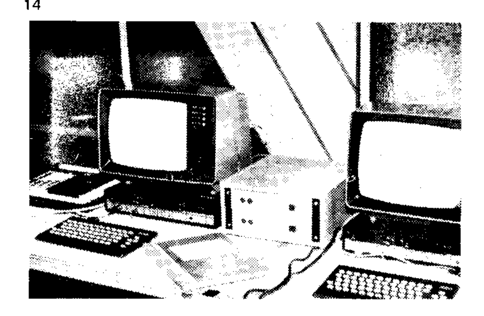
8



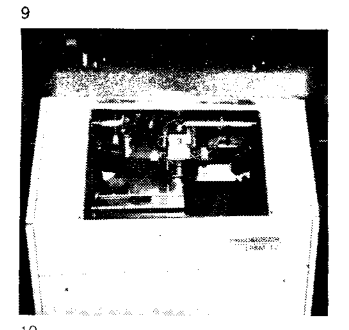
11



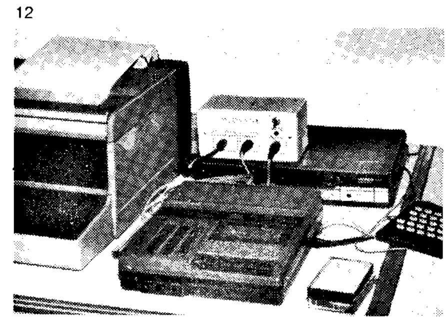
14



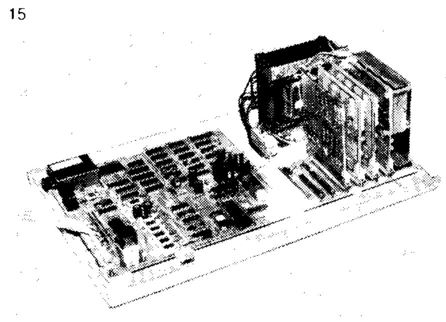
9



12



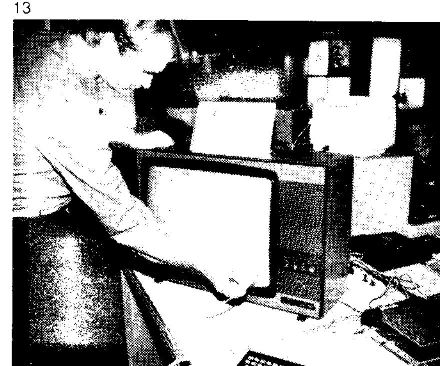
15



10



13



Auf dem waagerecht angeordneten Baugruppenträger können bis zu 4 Erweiterungskarten gesteckt werden (davon muß eine STE die Stromversorgungskarte sein; evtl. wird der Austausch des Netztrafos notwendig!). Als Erweiterung gibt es u. a. den ROM-Modul mit 4 Steckplätzen für EPROMs und einen E/A-Modul, u. a. mit V.24, aber auch der RAM-Modul des KC 87 ist z. B. nutzbar. (*VEB Robotron-Elektronik Riesa, Pausitzer Straße 60, Riesa, 8400*)

Text und Fotos: Weiß



Börse

RAM-Floppy-Treiber für A 5120/5130

Für das auf A 5120/5130 laufende Betriebssystem UDOS wurde ein RAM-Floppy-Treiber entwickelt. Damit steht dem Anwender ein effektives „Laufwerk" zur Verfügung, mit dem sich vor allem die Arbeitsgeschwindigkeit bei der Programmerstellung (EDIT, ASM, LINK) erheblich vergrößert. Die Organisation der RAM-Floppy und die Verwaltung (256 K) werden vollständig von ZDOS und dem Treiber übernommen. Am Betriebssystem erfolgten keinerlei Änderungen. Der RAM-Floppy-Treiber ist an das I/O-Request-Prinzip des UDOS-BS angepaßt und kann wie ein E/A-Gerät behandelt werden.
Die verwendete Speicherkarte wurde in Mikroprozessortechnik 3/87 beschrieben.
VEB RFT Nachrichtenelektronik Leipzig; Büro für Neuererwesen, Abt. TT3, Melscher Str. 7, Leipzig, 7027; Tel. 6832391 *Dr. Schmidt*

Programmentwicklung für U881/882 auf LC-80

Entwicklung und Test von Maschinenprogrammen für Einchip-Mikrorechner U881/882 sind an relativ teure Hardware gebunden. Als Alternative hierzu wurde deshalb ein Simulationsprogramm entwickelt, in welchem unter Beibehaltung des EMR-Adreßbereiches alle Befehle des EMR auf einem LC-80 abgearbeitet werden können. Notwendig ist lediglich eine Hardware-Erweiterung des LC-80 mit 2 EPROM 2716, versehen mit dem Simulationsprogramm, sowie 4 Stück RAM U214 auf den vorgesehenen Bestückungsplätzen.
Folgende Funktionen werden realisiert:
- Verwendung der Originaladressen des EMR (0000–07FF)
- Start des EMR-Programms an einer beliebigen Adresse
- Setzen beliebig vieler Haltepunkte im EMR-Programm
- Anzeige und Manipulierung beliebiger Registerinhalte an den Haltepunkten
- Einzelschrittbetrieb bei Anzeige dreier beliebiger Register
- langsamer (ca. 70 Befehle/s) Durchlaufbetrieb mit Tonausgabe je abgearbeitetem Befehl sowie Anzeige dreier Register
- schneller (ca. 400 Befehle/s) Durchlaufbetrieb ohne Anzeige
- Betriebsarten beliebig kombinierbar
- Zählung der abgearbeiteten Befehle zur überschläglichen Berechnung der Programmlaufzeit.

Folgende Einschränkungen sind vorhanden:
- Keine Ein- und Ausgaben externer Signale
- Portprogrammierung zwar möglich, aber ohne Wirkung
- kein Echtzeitbetrieb
- keine Interruptfähigkeit
- keine externe Programm- oder Datenspeicher-Erweiterung.

Das Simulationsprogramm ist somit für folgende Anwendungen geeignet:
- Intensives Kennenlernen des Befehlssatzes der EMR U881/882
- Entwicklung und Test von Programm-Moduln
- Entwicklung und Test komplexer Programme für EMR.

VEB Kaliwerk „Heinrich Rau", Roßleben, 4735; Tel. 205, App. 732 *Denkes*

K-1520-A/D-Wandler mit 16-Kanal-Multiplexereinheit

Der A/D-Wandler ist für den Einsatz in der mikrorechnergestützten Meßtechnik bzw. Prozeßdatenerfassung bestimmt; er ist komplett auf einer Leiterkarte in der Größe einer K-1520-Steckeinheit realisiert.
Der Wandler treibt einen vollständigen K-1520-Datenbus, belastet die Bussignale mit 1/6 TTL-Last (Low Power) und ist somit K-1520-buskompatibel.
Der Eingangsspannungsbereich des 16kanaligen Analogmultiplexers (Bestandteil der Baugruppe) beträgt ± 1 V. Somit ist die A/D-Wandlereinheit für die Erfassung kleiner Signalamplituden gut geeignet. Alle Eingänge sind vor Überspannungen geschützt.
Über Programmbefehle wird der Multiplexer frei wählbar geschaltet, durch die Kanalauswahl erfolgt gleichzeitig der Wandlerstart.
Die A/D-Wandlung wird mit dem Schaltkreis C 571 (AD 571) /C570 (AD 570) realisiert. Der Wandler setzt 11 (9) Bit einschließlich Vorzeichen um.
Die fertige Wandlung wird über einen Interrupt gemeldet. Die Umsetz- einschließlich Multiplexereinschwingzeit beträgt vom Start bis zum Interrupt bei Kanalwechselbetrieb 50 μs, bei Schnellstart ohne Kanalwechsel 30 μs. Die verwendete PIO-Schnittstelle ermöglicht die Nutzung sämtlicher Interruptmodi der U880-Technik.
Zum Nachnutzungsumfang der Lösung gehören die technische Dokumentation sowie eine unbestückte industriell gefertigte Leiterplatte.
Technische Hochschule Ilmenau, Sektion Technische und Biomedizinische Kybernetik, Wissenschaftsbereich Biomedizinische Technik und Bionik, PSF 327, Ilmenau, 6300; Tel. 740 *Prof. Dr. Hennig*

Grafikprogramme

Grafikzeichen für KC 85/2 und /3 selbstdefiniert

Zur vereinfachten Erstellung nutzerdefinierter Grafikzeichen für den KC 85/2 oder /3 wurde ein Programm entwickelt, das es ermöglicht, auf einfache Art und Weise direkt am Bildschirm die Zeichen zu generieren.
In die vergrößerte 8 × 8-Zeichenmatrix kann der Nutzer sein eigenes Zeichen eintragen. Das Programm ermittelt die Bytesequenzen für eine spätere programmtechnische Bereitstellung der Zeichen. Außerdem kann das kreierte Zeichen sofort dargestellt und genutzt werden.

Grafikprogramm für KC 85/2 und /3

Ein universelles Grafikprogramm in Unterprogrammtechnik ermöglicht die Darstellung von stetigen Funktionen mit beliebigem Wertebereich und die Darstellung diskreter Wertepaare als Quasifunktion oder Säulendiagramm.
Die Maßstäbe werden in Abhängigkeit vom Wertebereich und den dazugehörigen Argumenten automatisch gewählt. Konzipiert ist das Programm für die Einbindung der jeweiligen Module in ein beliebiges Hauptprogramm. Zu Demonstrationszwekken wurden beide Unterprogramme durch Eingabe- und Definitionsroutinen ergänzt. Damit ist das Programm auch in dieser Form lauffähig.
Hochschule für Ökonomie, Informatikzentrum der Sektion Außenwirtschaft, Herrmann-Duncker-Str. 8, Berlin, 1157 *Iffarth*

BASIC-Lernprogramme

Die Lernprogramme dienen einer ersten Einführung Ungeübter in die Programmiersprache BASIC, enthalten aber auch für Fortgeschrittene interessante Hinweise. Erläutert werden alle Befehle des KC 85/2 und /3, die im BASIC verfügbar sind. Außerdem wird auf das Betriebssystem CAOS näher eingegangen und das Speicherkonzept erläutert.
Der Anwender bekommt die Möglichkeit, parallel zu den Erklärungen auch üben zu können. Durch die Fenstertechnik bleiben dabei die Beispiele immer auf dem Bildschirm. Der Nutzer arbeitet in einem eigenen Fenster.
Hochschule für Ökonomie, Informatikzentrum der Sektion Außenwirtschaft, Herrmann-Duncker-Str. 8, Berlin, 1157 *Iffarth*

Dokumentation für C-Compiler

Dokumentationen in deutscher Sprache für den BDS-C-Compiler und ECO-C-Compiler unter CP/M-kompatiblen Betriebssystemen, vorzugsweise CPA mit 54 KByte TPA, einschließlich einer Übersicht zur jeweiligen Standortbibliothek werden zur Nachnutzung auf Diskette angeboten.
Gesucht werden Partner zum Erfahrungsaustausch zu Problemen des Microsoft-C-Compilers ab Version 4.0 unter MS-DOS.
VEB Kombinat Anlagen- und Gerätebau Halle, Direktionsbereich Forschung/Entwicklung, Abteilung Automatisierungstechnik, Straße der DSF 86a, Halle, 4014; Tel. 8330, App. 414 oder 419 (Koll. Hauk, Koll. Dr. Scharfe) *Dr. Beese/Dr. Scharfe*

Abrechnung des PWT und Neuererwesens

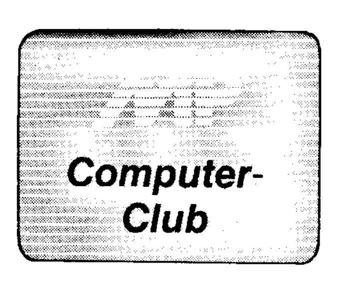
Das Programm in TURBO-PASCAL ist anwendbar bei allen BC/PC, die ein CP/M-kompatibles Betriebssystem besitzen. Eine Programmvariante für den AC 7100 wird zur Zeit erarbeitet. Es ist eine Diskette mit Software und eine zweite für die Daten vorgesehen.
Die ökonomischen Ergebnisse des PWT werden nach folgenden Kriterien abgerechnet:
- Planteile
- Urheberkostenstellen
- Wirkungsbereiche
- Charakteristiken (z. B. MMM, Nachnutzung, Mikroelektronik usw.)
- Materialarten (Menge und Wert)

Ausgewählte Schwerpunktmaterialien werden mengenmäßig auf Urheberkostenstellen aufgeschlüsselt. Es wird jeweils ein Soll/Ist-Vergleich durchgeführt und die prozentuale Erfüllung ausgewiesen.
Die Abrechnungszeiträume umfassen:
- Ist im Berichtszeitraum
- Ist für beliebigen Folgemonat
- Ist per Jahresende

und Überhang.
Als Darunterposition des PWT werden die Ergebnisse des Neuererwesens in gleicher Art und Weise abgerechnet und ausgewiesen, ohne daß eine Neueingabe erfolgt.
Des weiteren sind einige Serviceroutinen implementiert. Der gesamte Ablauf des Programmes wird mit Hilfe von Menüs gesteuert. Zum Programm gehört ein Installationsprogramm, mit dem es möglich ist, die genannten Abrechnungsarten inhaltlich und im Umfang zu ändern, ohne daß eine Neuübersetzung des Programms notwendig ist.
VEB Elektromaschinenbau Dresden, Elektromotorenwerk Dresden-Nord, Abt. TN, Großenhainer Str. 109, Dresden, 8023; Tel. 52511, App. 216 *Dittrich/Weißfloh*

Beliebige PC an SKR-Rechner gekoppelt

Mehrplatzsysteme des SKR und der zunehmende dezentrale Einsatz von 8- und 16-Bit-PC wie BCA 51xx, PC 1715, AC 71xx, PC 183x, AKTS sowie AT- und XT-kompatibler Geräte führen zur Forderung nach bequemem und zuverlässigem Datenaustausch zwischen all diesen Geräten.
Mit der nun auch für 16-Bit-PC (und AC) bereitstehenden Software TERMINAL wird dieses Problem unter Nutzung der Fähigkeit der SKR-Rechner zur simultanen Bedienung mehrerer Terminals über serielle Verbindungen (V24, IFSS) zuverlässig und komfortabel gelöst. Da es sich um Software unterhalb des Niveaus von Rechnernetzen handelt, entsteht auch kein Aufwand für Hardware.
Mit TERMINAL ist es möglich, alle gebräuchlichen 8- oder 16-Bit-PC als VT100-kompatible aktive Terminals nah- oder fernaufgestellt in Kopplung mit SKR-Anlagen oder entsprechenden 32-Bit-Mehrplatzsystemen zu betreiben. Zusätzlich ist Dateiaustausch ohne Einschränkungen des Datenformats in jeder Richtung möglich. Diese Funktion wird auch über eine Softwareschnittstelle bereitgestellt (Include-Prozedur).
TERMINAL läuft unter SCP, SCP 1700, DCP sowie unter den zu CP/M, CP/M-86 und MS-DOS kompatiblen Betriebssystemen.
VE Kombinat Baureparaturen und Rekonstruktion, ORZ, Markt 8, Leipzig, 7010; Tel. 209688, 298824 *Dr. Siebach*



Entfernen von REM-Zeilen

Das Dokumentieren von Quelltext sollte für jeden Programmierer zur Selbstverständlichkeit gehören. Leider belegen Kommentarzeilen in einem BASIC-Programm teilweise mehr als die Hälfte des beanspruchten Speicherplatzes. In der Arbeitsversion des Programms sind diese Zeilen überflüssig. Das vorliegende Programm ist ein nützliches Werkzeug zum Löschen dieser Kommentarzeilen.

Da beim KC 85/3 der DELETE-Befehl zum Abbruch führt, wenn er innerhalb eines Programms angewendet wird, wurde die in /1/ veröffentlichte Idee genutzt, um eine weitere Programmerarbeitung zu erreichen.

Weiterhin muß die Nummer der zu löschenden Zeile hinter dem Token für DELETE im ASCII-Code in den Speicher geschrieben werden.

Das entsprechende Programm ist in Abb. 1 zu sehen. Das Programm wird mittels LOAD#1 geladen und fügt sich vor dem zu bearbeitenden Programm ein, welches erst mit Zeilennummer 20 beginnen darf. Es liest jeden Speicherplatz vom Programmanfang bis Programmende und vergleicht dessen Inhalt mit den Token für 'REM' bzw. '!'. Wird ein entsprechendes Byte gefunden, wird außerdem auf Zeilenanfang verglichen, um auszuschließen, daß das Byte Teil einer Adresse oder Zeilennummer ist. Danach wird die zugehörige Zeilennummer gelesen, in ASCII-Code umgewandelt und in den Speicher geschrieben. Nach dem Löschen der Zeile wird das Programm mit der GOTO-Anweisung erneut aktiviert. Die aktuelle Speicheradresse für den Vergleich steht dabei immer in Adresse OAH. Jede gelöschte Zeile wird auf dem Bildschirm angezeigt.

Wurde das gesamte Programm durchsucht, werden die Programmzeilen des REM-"Killers" gelöscht. Damit liegt die Arbeitsversion des bearbeiteten Programms vor.

Kai Hauser

Hinweis der Redaktion:

Das Programm muß genau zeichenweise eingegeben werden, ohne dessen Länge z. B. durch Leerzeichen zu verändern. Außerdem darf die letzte Programmzeile im zu bearbeitenden Programm keine Kommentarzeile sein, das sonst das abgedruckte Programm kein Ende findet.

Literatur

/1/ Kirves, K.-D.: Arbeit mit BASIC-Datenfeldern beim KC 85/3. Mikroprozessortechnik 1 (1987) 3, S. 94

```
2 CLS:PRINT"REM-KILLER"
3 DOKE10,DEEK(863)
4 A¤="GOTO6"+CHR¤(13)+CHR¤(0)
5 FORX=1TO7:POKEX,ASC(MID¤(A¤,X,1)):NEXT
6 POKE504,PEEK(504)OR64:VPOKE14289,1 VPOKE14290,0
8 FORZ=0TO4:POKE1431+Z,48:NEXT
9 C=DEEK(10):BY=PEEK(C):C=C+1:IFC=DEEK(983)GOTO11
10 IF(BY=156ORBY=142)ANDPEEK(C-6)=0GOTO12:ELSEDOKE10,C:GOTO9
11 POKE504,PEEK(504)AND191:DELETE2,18
12 ZN=PEEK(C-2)*256+PEEK(C-3):C=C-2:DOKE10,C
13 Z¤=STR¤(ZN):L=LEN(Z¤):Z¤=RIGHT¤(Z¤,L-1):L=L-1
14 PRINT"ZN=";ZN,
15 FORG=LTO1STEP-1
16 Z=VAL(MID¤(Z¤,G,1)):POKE1435-L+G,Z+48:NEXT
18 DELETE00000
```

Programmieren der Funktionstasten am PC 1715 in BASIC

Der Personalcomputer PC 1715 ist mit einer relativ komfortablen Tastatur ausgestattet. Leider ist mit der vorhandenen Dienstleistungssoftware unter dem Betriebssystem SCP keine Möglichkeit geschaffen worden, eine schnelle Veränderung der Funktionstastenprogrammierung zu realisieren. Die Programmierung erfolgt in der Regel mit dem Hilfsprogramm INSTSCP, wobei es dem wenig erfahrenen Nutzer mit der hexadezimalen Verschlüsselung des Buchstaben- und Tastencodes schwer gemacht wird. Durch das BASIC-Programm wird der entsprechende Adreßraum von E0AA–E0EB (Dezimal 57514–57579) gelesen und auf einfache Form wieder neu beschrieben.

Zum besseren Verständnis des Programms ist jedoch der rechnerinterne Aufbau dieses Adreßraumes von Bedeutung. Die aktuellen Tastencode sind beispielsweise nach folgendem Schema abgespeichert:

Hex. Adresse	Dez. Adresse	Abgelegter Code (Hex)	Bedeutung
E0AA	57514	D1	Nummer der Taste F1 /1/
E0AB	57515	04	Anzahl der programmierten Code
E0AC	57516	52	ASCII-Code für R
E0AD	57517	55	ASCII-Code für U
E0AE	57518	4E	ASCII-Code für N
E0AF	57519	0D	ASCII-Code für <ET>
E0B0	57520	D4	Nummer der Taste F4
E0B1	57521	01	Anz. der progr. Zeichen
E0B2	57522	19	ASCII-Code für ctrl S
E0B3	57523	83	Nummer der Taste F10
...	...	..	....
E0EB	57579	FF	Endekennzeichen

(Die Anordnung läßt sich mit dem Programm POWER und dem Kommando DUMPX E0A0 E0EF überprüfen.)

Dabei spielt die Reihenfolge und die Anzahl der programmierten Tasten keine Rolle. Jedoch können insgesamt nur 66 Code abgespeichert werden. Für jede beliebige Tastencodierung gilt, daß pro Taste mindestens 3 Code belegt werden (siehe Adresse E0B0 bis E0B2).

Dagobert Mühlhaus

Literatur

/1/ Manual robotron 1715, Pkt. 2.2.3.

```
10 DEFINT A,B,I,J,Z,Y,C:DIM A(7Ø)
2Ø DATA 2Ø9,21Ø,211,212,2Ø7,16Ø,161,162,163,131,193,192,194,2Ø5,142
3Ø '     F1 ,F2 ,F3 ,F4.,F5 ,F6 ,F7 ,F8 ,F9 ,F1Ø,F11,F12,F13,F14,F15
4Ø PRINT CHR$(12)"      Funktionstastenprogrammierung"
5Ø PRINT"      "+STRING$(31,"=")
6Ø PRINT"    Lesen der Belegung aus dem Hauptspeicher --> 1"
7Ø PRINT"    Aendern der Belegung"TAB(45)"--> 2"
8Ø PRINT"    Anzeigen der Tastenbelegung"TAB(45)"--> 3"
9Ø PRINT"    Sichern der neuen Belegung im Hauptsp.   --> 4"
1ØØ PRINT"    Laden von Diskette"TAB(45)"--> 5"
11Ø PRINT"    Sichern auf Diskette"TAB(45)"--> 6"
12Ø PRINT"    Ende"TAB(45)"--> 7":INPUT"Bitte waehle Sie :",CODE:I=1
13Ø PRINT CHR$(12):RESTORE:ON CODE GOSUB 15Ø,24Ø,18Ø,36Ø,39Ø,43Ø,49Ø
14Ø INPUT"Weiter mit <ENTER>";XX:GOTO 4Ø
15Ø '=============Lesen der Funktionstastenbelegeung (Hauptspeicher)
16Ø A(I)=PEEK(I+57513!):IF A(I)=255 THEN RETURN:ELSE I=I+1:GOTO 16Ø
17Ø '=============Anzeige der Tastenbelegung (neu / alt)
18Ø IF A(I)=255 THEN RETURN
19Ø FOR J=1 TO 15:READ B:IF A(I)=B THEN 2ØØ:ELSE NEXT J:J=Ø
2ØØ RESTORE:PRINT"     F"J" ===> ";:FOR I=I+2 TO I+1+A(I+1)
21Ø IF A(I)<32 THEN PRINT"^"CHR$(A(I)+64);:ELSE PRINT CHR$(A(I));
22Ø NEXT I:PRINT:GOTO 18Ø
23Ø '=============Tastenprogrammierung
24Ø PRINT"Ende Tastenprogrammierung mit '/'
25Ø Z=3:Y=Ø: FOR I=1 TO 15:PRINT"  F"I"==>";:READ B:A(Z-2)=B
26Ø C$=INPUT$(1)
27Ø IF C$="/" THEN IF Y=Ø THEN 33Ø: ELSE A(Z-Y-1)=Y: GOTO 32Ø
28Ø A(Z)=ASC(C$)
29Ø IF A(Z)<32 THEN PRINT"^"CHR$(A(Z)+64);:ELSE PRINT CHR$(A(Z));
3ØØ Z=Z+1:Y=Y+1:IF Z>66 THEN PRINT"Speicher voll":GOTO 34Ø
31Ø GOTO 26Ø
32Ø Y=Ø:Z=Z+2
33Ø PRINT:NEXT I
34Ø A(Z-2-Y)=255:RETURN
35Ø '=============Sichern neue Belegung im Hauptspeicher
36Ø POKE 57513!+I,A(I):IF A(I)=255 THEN CLOSE:RETURN
37Ø IF A(I)=255 THEN RETURN:ELSE I=I+1:GOTO 36Ø
38Ø '=============Holen Tastenbelgung von Datei *.FUT
39Ø GOSUB 47Ø
4ØØ GET#1:A(I)=VAL(CO$)
41Ø IF A(I)=255 THEN CLOSE#1:RETURN:ELSE I=I+1:GOTO 4ØØ
42Ø '=============Ablegen Tastenbelegung auf Datei *.FUT
43Ø GOSUB 47Ø
44Ø LSET CO$=STR$(A(I)):PUT#1
45Ø IF A(I)=255 THEN CLOSE#1:RETURN:ELSE I=I+1:GOTO 44Ø
46Ø '=============Dateieröffnung für *.FUT
47Ø INPUT"Dateiname (LW: und max. 8 Buchstaben) ";NA$
48Ø OPEN"R",#1,NA$+".FUT",4:FIELD#1,4 AS CO$:RETURN
49Ø END
```

Nochmals: MC-Code und BASIC

Neben den in MP 10/87 vorgestellten Möglichkeiten der Einbindung von MC-Code in BASIC-Programmen kann es praktisch sein, den Maschinenteil vor den BASIC-Beginn zu legen. Die BASIC-Programme lassen sich dann erweitern und editieren. Das komplette Programm wird ab 035FH oder noch besser ab 0300H mit ⁄SAVE gerettet. Die Endadresse steht in 03D7H/03D8H.

Das Verlegen des Programmspeichers erfolgt nach BASIC-Kaltstart und Rückkehr in das Betriebssystem. In 035FH/0360H wird mit MODIFY der Zeiger auf die erste BASIC-Zeile gestellt (Original: 0401H)

⁄REBASIC und >NEW sorgen für die notwendigen Einträge im Notizspeicher.

Ist das Programm fertig, kann es mit einer Selbststartadresse abgespeichert werden /1/.

Uwe Zierott

Literatur

/1/ Zierott, U.: Erweiterte Zeilenbefehle für den KC-BASIC-Interpreter. Mikroprozessortechnik 1 (1987) 11, S. 333–334

Termine

2. zentraler Erfahrungsaustausch von Computerklubleitern

WER? Bezirksvorstand der KDT Potsdam

WANN? Mai 1988

WO? Potsdam

WAS? Erfahrungsaustausch zu folgenden Schwerpunkten:
- Stand und Entwicklung der technischen Basis
- didaktische Probleme
- Organisation der Klubarbeit
- Aufgaben und gesellschaftspolitische Rolle der Betreiber individueller Rechentechnik

WIE? Teilnahmemeldungen bitte an Bezirksvorstand Potsdam der Kammer der Technik, Weinbergstraße 20, PSF 27, Potsdam, 1561

Rand

Pixelvergrößerung für KC 85/2 (/3)

Das Unterprogramm wurde entwickelt, um Programme einem größeren Interessentenkreis mit einem Monitor vorführen zu können. Es wurden sowohl neue Programme mit diesem UP geschrieben als auch bereits vorhandene Programme mittels dieses UPs auf Großdarstellung umgerüstet (Änderung der Monitorgestaltung, aber keine Änderung des Rechenteils vorhandener Programme).

Folgende Varianten sind z. B. möglich:

– Wahlweise Klein- oder Großdarstellung je nach Vorführzweck des Programmes (über IF-THEN-Anweisungen realisierbar)
– Reine Großdarstellungen (Steuerstrings "SA" und "SL") der Druckbefehle
– Normallauf von Programmen und Großdarstellung der aktuell interessierenden Druckzeile zusätzlich an oberem oder unterem Monitorrand (Steuerstring "SA").

Sonderzeichen, die vom Anwender über CCTL-Adressierung definiert sind, werden vom UP ebenfalls vergrößert. Es ist auch möglich, Groß- und Kleindarstellungen nebeneinander (z. B. für Exponenten und Indizes) zu verwenden.

Großdarstellung von Strings, die länger als eine Druckzeile sind, ist über das Hauptprogramm zu organisieren.

Um den Umgang mit dem UP zu erleichtern, wurden umfangreiche Regieanweisungen (ab 58000) und ein Testprogramm (ab 59000) dem eigentlichen UP (ab 60000) vorgeschaltet.

Beim erstmaligen Anspringen des UP wird der MC-Teil des UP geladen (60030–60330), deshalb dauert in diesem Fall die Vergrößerung eines Strings länger als bei den folgenden UP-Aufrufen.

Das UP wurde auf dem KC 85/2 (/3) mit BASIC-Interpreter-Version 2.13/1) getestet. Die vorliegende Variante des UP ist der Stand 9/87.

Dr. Gunter Bonitz

✉ KONTAKT ☎

Technische Universität Karl-Marx-Stadt, Sektion Vorkurse, Dr. G. Bonitz, Ernst-Schneller-Straße 128, Breitenbrunn, 9434


```
58ØØØ ! HINWEISE ZUM UP PIVE FUER KC2/3
58Ø1Ø ! ===============================
58Ø2Ø !
58Ø3Ø ! DAS UP GESTATTET,MIT DOPPELTER
58Ø4Ø ! PIXELGROESSE ALLE ZEICHEN UND
58Ø5Ø ! SONDERZEICHEN DARZUSTELLEN.
58Ø6Ø ! DAMIT IST DER MONITOR BIS IN CA
58Ø7Ø ! 15 METER ENTFERNUNG NOCH GUT
58Ø8Ø ! LESBAR, SO DASS ENTSPRECHENDE
58Ø9Ø ! PROGRAMME EINEM GROSSEN INTER-
581ØØ ! ESSENTENKREIS VORGEFUEHRT WER-
5811Ø ! DEN KOENNEN.
5812Ø ! DAS UP VERWENDET ZWEI MASCHI-
5813Ø ! NENPROGRAMME MP1 (2ØH-31H) UND
5814Ø ! MP2 (BD8ØH-BF6ØH),EINGELESEN
5815Ø ! IM UP VON 6ØØ3Ø BIS 6Ø34Ø.
5816Ø ! BASIC-RAM-SPEICHERBEDARF:
5817Ø ! UP AB 6ØØØØ:CA 2 KBYTE
5818Ø ! UP AB 6Ø35Ø:CA Ø.2 KBYTE
5819Ø !       (MP1/2 DIREKT EINGELESEN)
582ØØ !
5821Ø ! PARAMETERUEBERGABE AN DAS UP
5822Ø ! ----------------------------
5823Ø ! ZP$ :ZU VERGROESSERNDER TERM
5824Ø ! ZC$ :STEUERSTRING,SIEHE LISTE
5825Ø ! ZZ  :ZEILE DES DRUCKBEGINNS
5826Ø ! ZS  :SPALTE DES DRUCKBEGINNS
5827Ø !
5828Ø ! FUER DIE UEBERGABE SIND DIE
5829Ø ! SPEICHERPLAETZE BDØØH-BD7FH
583ØØ ! RESERVIERT.
5831Ø !
5832Ø ! STEUERSTRINGS SIND "PN","PL",
5833Ø !     "PA","IN","IL","SL","SA".
5834Ø ! P :VERGROESSERUNG NACH PRINT-
5835Ø !    BEFEHLEN AN DIESEM ORT
5836Ø ! I :VERGROESSERUNG NACH INPUT-
5837Ø !    BEFEHLEN AN DIESEM ORT
5838Ø ! S :VERGROESSERUNG VON STRINGS
5839Ø !    AN BELIEBIGEM ORT
584ØØ ! N :NORMALAUSGABE UND -EINGABE
5841Ø ! L :LOCATEAUSGABE UND -EINGABE
5842Ø ! A :AT-AUSGABE
5843Ø !
5844Ø ! ANGABE VON ZZ UND ZS IST ERFOR-
5845Ø ! DERLICH FUER FOLGENDE FAELLE:
5846Ø !
5847Ø ! ZC$: PN  PL  PA  IN  IL  SL  SA
5848Ø ! ZZ :                      *   *
5849Ø ! ZS :                  *   *   *
585ØØ !
5851Ø ! AUS- UND EINGABESTEUERZEICHEN
5852Ø ! WIE "SPC", "TAB", ",", ";"
5853Ø ! KOENNEN DURCH DAS UP NICHT BE-
5854Ø ! ARBEITET WERDEN.
5855Ø ! DURCH DAS UP WERDEN DIE LETZTE
5856Ø ! ZEILE UND SPALTE DES AKTUELLEN
5857Ø ! FENSTERS NICHT BEDRUCKT.
5858Ø !
5859Ø ! EIN TESTPROGRAMM AB 59ØØØ ZEIGT
586ØØ ! DIE WIRKUNGSWEISE DES UP.
5861Ø !
5862Ø ! VOR ANWENDUNG DES UP KOENNEN
5863Ø ! DIE ZEILEN 58ØØØ BIS 5999Ø
5864Ø ! GELOESCHT WERDEN.
5865Ø !
59ØØØ WINDOWØ,31,Ø,39:COLOR6,1:CLS:WINDOW2,29,3,35:COLOR1,6 :CLS
59Ø1Ø PRINT:INPUT"X=";X:ZC$="IN":ZP$="X="+STR$(X):GOSUB6ØØØØ
59Ø2Ø PRINTX:PAUSE(2Ø):ZP$=STR$(X):ZC$="PN":GOSUB6ØØØØ
59Ø3Ø WINDOW7,28,4,34:COLOR6,2:INPUT"Y$=";ZP$:ZC$="IN":GOSUB6ØØØØ
59Ø4Ø ZP$="Y$="+ZP$:PRINTAT(17,Ø);ZP$:ZC$="PA":GOSUB6ØØØØ
59Ø5Ø LOCATE15,2Ø:ZS=2Ø:INPUTX:ZC$="IL":ZP$=STR$(X):GOSUB6ØØØØ
59Ø6Ø LOCATE13,15:PRINT"PQ=";3:ZC$="PL":ZP$="PQ= 3 ":GOSUB6ØØØØ
59Ø7Ø PRINTPI:ZZ=29:ZS=1:ZC$="SA":ZP$=STR$(PI):GOSUB6ØØØØ
59Ø8Ø WINDOW5,25,1Ø,3Ø:ZP$="Aa"+CHR$(193)+CHR$(225):ZC$="SL":ZZ=4:ZS=3
59Ø9Ø GOSUB6ØØØØ
591ØØ ZC$="SA":ZP$=CHR$(141):ZZ=29:ZS=37:GOSUB6ØØØØ
5999Ø END
6ØØØØ IFVPEEK(15631)THENGOTO6Ø35Ø:ELSERESTORE6ØØ3Ø
6ØØ1Ø !UP PIXELVERGROESSERUNG KC2/3
6ØØ2Ø !AUTOR:G.BONITZ,TUK/BREITENBRUNN
6ØØ3Ø DATA245,219,136,2Ø3,215,211,136,2Ø5,135,189,219,136,2Ø3
6ØØ4Ø DATA151,211,136,241,2Ø1
6ØØ5Ø FORZK=32TO49:READZX:POKEZK,ZX:NEXT
6ØØ6Ø DATA127,127,8Ø,73,86,69,1,229,213,197,245,253,229,58,1,189,254,65
6ØØ7Ø DATA32,28,1,6,Ø,33,156,183,17,5,189,237,176,175,5Ø,156,183,5Ø
6ØØ8Ø DATA157,183,62,4Ø,5Ø,158,183,62,32,5Ø,159,183,58,Ø,189,71,62,8Ø
6ØØ9Ø DATA184,32,61,58,4,189,79,253,33,7,192,253,126,Ø,254,66,32,6
6Ø1ØØ DATA253,33,152,1,24,28,42,86,42,17,82,Ø,175,186,237,82,58,1
6Ø11Ø DATA189,254,65,4Ø,4,17,12,Ø,25,34,94,189,253,42,94,189,253,126
6Ø12Ø DATAØ,6Ø,145,111,253,126,1,1Ø3,24,29,62,73,184,32,16,58,161,183
6Ø13Ø DATA61,1Ø3,58,1,189,254,76,4Ø,8,175,111,24,8,58,3,189,1Ø3,58
6Ø14Ø DATA2,189,111,34,16Ø,183,1,16,189,237,67,13,189,235,2Ø,2Ø,28,28,2Ø5
6Ø15Ø DATA3,24Ø,51,218,54,191,237,75,13,189,1Ø,79,214,32,95,22,Ø,2Ø3
6Ø16Ø DATA35,2Ø3,18,2Ø3,35,2Ø3,18,2Ø3,35,2Ø3,18,121,254,96,48,6,42,166
6Ø17Ø DATA183,25,24,47,254,128,48,7,42,168,183,25,37,24,36,254,16Ø,48
6Ø18Ø DATA9,42,172,183,25,37,37,37,24,23,254,224,48,1Ø,42,17Ø,183,25
6Ø19Ø DATA37,37,37,37,24,9,42,172,183,25,37,37,37,37,37,34,11,189
6Ø2ØØ DATA253,42,11,189,253,229,253,33,96,189,14,Ø,253,227,253,126,Ø,253
6Ø21Ø DATA227,6,8,17,3,Ø,33,Ø,Ø,2Ø3,71,4Ø,1,25,2Ø3,35,2Ø3,18
6Ø22Ø DATA2Ø3,35,2Ø3,18,2Ø3,63,16,239,253,116,Ø,253,116,1,253,117,8,253
6Ø23Ø DATA117,9,62,3,161,254,3,4Ø,13,12,253,227,253,35,253,227,253,35
6Ø24Ø DATA253,35,24,192,62,252,161,198,4,254,8,4Ø,14,79,253,227,253,35
6Ø25Ø DATA253,227,17,1Ø,Ø,253,25,24,169,42,166,183,253,225,229,33,Ø,188
6Ø26Ø DATA34,166,183,62,76,2Ø5,3,24Ø,Ø,58,161,183,6Ø,5Ø,161,183,62,79
6Ø27Ø DATA2Ø5,3,24Ø,Ø,58,16Ø,183,61,61,5Ø,16Ø,183,62,78,2Ø5,3,24Ø,Ø
6Ø28Ø DATA58,161,183,61,5Ø,161,183,62,77,2Ø5,3,24Ø,Ø,225,34,166,183,58
6Ø29Ø DATA4,189,61,254,Ø,4Ø,16,5Ø,4,189,42,13,189,35,34,13,189,42
6Ø3ØØ DATA16Ø,183,195,29,19Ø,58,161,183,6Ø,6Ø,5Ø,161,183,175,5Ø,16Ø,183
6Ø31Ø DATA58,1,189,254,65,32,11,1,6,Ø,33,5
6Ø32Ø DATA189,17,156,183,237,176,253,225,241,193,2Ø9,225,2Ø1,2Ø,6,71,66,114,21
6Ø33Ø FORZK=15744TO16224:READZX:VPOKEZK,ZX:NEXT
6Ø34Ø VPOKE15631,1
6Ø35Ø VPOKE15616,ASC(LEFT$(ZC$,1)):VPOKE15617,ASC(RIGHT$(ZC$,1))
6Ø36Ø VPOKE15619,ZZ:VPOKE15618,ZS:ZN=LEN(ZP$):VPOKE1562Ø,ZN
6Ø37Ø FORZK=1TOZN:VPOKE15631+ZK,ASC(MID$(ZP$,ZK)):NEXT
6Ø38Ø CALL * 2Ø
6Ø39Ø RETURN
```

Ausdrucken von Bildschirminhalten beim PC 1715 unter SCP

Wird auf dem Personalcomputer PC 1715 in der Programmiersprache BASIC Software erstellt, ist es oft erforderlich, bestimmte Rechenergebnisse o. ä., die auf dem Bildschirm dargestellt werden, mit Hilfe des Druckers auszugeben.

Das erfordert aber doppelten Programmieraufwand, da die Befehle jeder PRINT-Anweisung noch einmal hinter einer LPRINT-Anweisung erscheinen müssen. Mit folgendem kleinen Unterprogramm, das sich durch die GOSUB-Anweisung an beliebiger Stelle im Programm einbauen läßt, fallen alle doppelten LPRINT-Anweisungen weg.

```
10 'HARDCOPY PC 1715
20 '############
30 'Ausdrucken Bildschirminhalt
40 WIDTH LPRINT 81
50 FOR I = 63488! TO 65423!
60    LPRINT CHR$(PEEK(I));
70 NEXT I:RETURN
```

Erläuterung des Programms

Der entsprechende Teil des Bildwiederholspeichers ist auf den Adressen F800H bis FF8FH abgelegt. Das entspricht dezimal den Adressen 63488 bis 65423. Er ist fortlaufend aufgebaut. Die Adresse 63488 entspricht der ersten Zeile und der ersten Spalte. Eine spezielle Buchstabenposition auf dem Bildschirm läßt sich ermitteln mit der Berechnung von:

POSITION = 63488 + 80 ∗ (Zeile – 1) + Spalte

Durch die Anweisung in Zeile 40 fällt eine zusätzliche Zählschleife für die schon gedruckten Zeichen pro Zeile weg.

Besondere Effekte auf dem Bildschirm kann man durch direktes Beschreiben des Bildwiederholspeichers möglich machen. Das kann man problemlos mit der POKE-Anweisung in der Form: POKE POSITION,BYTE realisieren, wobei das gewählte Byte ab der gesetzten Position bis Bildschirmende gilt. In folgendem Beispiel seien einmal ein paar Code ausgewählt.

```
10 PRINT CHR$(12)
20 FOR I = 0 TO 6:READ CODE
30 PRINT
   "ABCDEFabcdef12345∗+⟨⟩"
40 POKE
   63488+I∗80,CODE:NEXT I
50 DATA
   133,135,160,130,131,144,145
```

Es wurde jeweils die erste Spalte mit einem Byte belegt.

Wirkungen der Code der DATA-Anweisung:

133 sehr heller Hintergrund, keine Schrift
135 sehr heller Hintergrund, dunkle blinkende Schrift
160 Durchgehende Linie, dunkler Hintergrund
130 Dunkler Hintergrund, helle blinkende Schrift
131 Dunkler Hintergrund, sehr helle blinkende Schrift
144 Heller Hintergrund, dunkle Schrift
145 Sehr heller Hintergrund, dunkle Schrift

Dagobert Mühlhaus

Grundlagen und Anwendungen der CAD/CAM-Technologie – eine Fachbibliographie

von einem Autorenkollektiv, Heft 17 der Schriftenreihe der Betriebssektion der KDT und der Zentralen Informationsstelle Wissenschaft und Technik des VEB GRW „Wilhelm Pieck" Teltow, 1987, 70 S., 8,– M

Die rechnergestützte Konstruktion/ Projektierung und Fertigung gewinnt zunehmend entscheidende Bedeutung bei der Nutzung solcher Schlüsseltechnologien wie Mikroelektronik, Sensortechnik, Mikrorechentechnik, flexible automatisierte Fertigungssysteme, Optoelektronik, Lasertechnik und Anwendung neuer Werkstoffe.
Entsprechend dieser Grundrichtung der wissenschaftlich-technischen Entwicklung sind dieser Fachbibliographie die neuesten Veröffentlichungen auf dem Gebiet der rechnergestützten Schlüsseltechnologien von CAD/CAM, CAE, CAQ bis hin zu CIM und vor allem der Entwicklung und Anwendung dieser rechnergestützten Technologien in den vielfältigsten industriellen Einsatzgebieten der Volkswirtschaft zugrunde gelegt. Zu diesen Einsatzgebieten zählen insbesondere die chemische Industrie, Energietechnik, Maschinenbau, Bauwesen, Elektrotechnik/Elektronik und die Automatisierungstechnik. Die Fachbibliographie enthält vorwiegend CAD/CAM-Anwendungsfälle, um den Entwicklern, Konstrukteuren und Technologen sowie den Kadern aller Leitungsebenen der Industrie sowie den in der Weiterbildung Tätigen Impulse für den Einsatz der CAD/ CAM-Technologie zu vermitteln. Die Literaturhinweise beziehen sich in erster Linie auf die Fachliteratur des deutschsprachigen Raumes. Die Fachbibliographie wird vervollständigt durch ein Sachwortverzeichnis, das auf die Ordnungsnummer des jeweiligen Literaturnachweises hinweist.
Mit dieser Publikation wird eine Lücke im derzeitig verfügbaren Fachliteraturangebot geschlossen.
Die Broschüre, die 490 Literaturnachweise enthält, die in begründeten Fällen mit Kurzreferat versehen sind, kann über die Zentrale Informationsstelle Wissenschaft und Technik bzw. über die Betriebssektion der Kammer der Technik des VEB GRW „Wilhelm Pieck" Teltow zum Preis von 8,– M bezogen werden. *Lothar Blackert*

Digitalgrafik

von Fritz Bulla, Akademie Verlag Berlin 1987, 213 S., 45,– M

Mit dem verstärkten Einsatz digitaler Rechentechnik in allen Bereichen der Gesellschaft verändern sich die Inhalte einer Vielzahl von Aufgabengebieten grundlegend. Diese Veränderungen dokumentieren sich nicht allein darin, daß diese Aufgaben den Zusatz „rechnergestützt" erhalten.
Die Mehrzahl dieser wissenschaftlichen und technischen Problemstellungen sind mit der Eingabe, Abbildung, Speicherung, Darstellung und Manipulation grafischer Informationen in geeigneter Form, z. B. als geometrisches Modell, verbunden. Diesem komplexen Aufgabengebiet, der Digitalgrafik bzw. grafischen Datenverarbeitung, widmet sich der Autor. Er stützt sich dabei auf Erfahrungen mit dem Gerätesystem GD 71 – KRS 4201 – ESER-Anlage. Den Nutzern dieser Konfiguration und Anwendern, die sich in die Problematik der Digitalgrafik einarbeiten, ist das vorliegende Buch sehr zu empfehlen.
In den ersten beiden Kapiteln werden in kurzer und übersichtlicher Form Grundlagen zu geometrischen Modellen und formalisierten Beschreibungen geometrischer Strukturen behandelt.
Die Beschreibung allgemeiner Funktionsprinzipien grafischer Ein- und Ausgabegeräte, wie z. B. Digitalisiergeräte, Plotter und Displays, folgt in Kapitel 3. Die betrachteten Beispiele beschreiben periphere Geräte des AKT 6454 oder von ESER-Anlagen. Der internationale Stand und die Entwicklungstendenzen werden analysiert.
Die Programmierung und Basissoftware des Zeichenautomaten DIGIGRAF werden in Kapitel 4 behandelt.
Den breitesten Raum nimmt Kapitel 5 mit der Programmierung von Bildschirmgeräten ein. Nachdem das Prinzip des Aufbaus von Bilddateien allgemein eingeführt wurde, wird eine konkrete Struktur am Beispiel einer Bilddatei des Gerätesystems GD 71/ KRS 4201 dargestellt. Daran anschließend folgt die FORTRAN-Programmierung zur passiven und interaktiven Nutzung von Bildschirmgeräten.
In Kapitel 6 werden die Arbeit am grafischen Terminal der Konfiguration GD 71 – KRS 4201 – ESER-Anlage und das zugehörige Programmsystem GIS/TEBAM behandelt.
Wichtig für Anwender und Programmierer von Grafiksoftware ist die in Kapitel 7 beschriebene Standardisierung der grafischen Schnittstelle. Es werden die FORTRAN-Funktionen des international standardisierten grafischen Kernsystems (GKS) beschrieben.
In den letzten drei Kapiteln beschäftigt sich der Autor mit der Ein- und Ausgabe geometrischer Informationen ohne größeren Programmieraufwand, speziellen Problemen der rechnerinternen Darstellung, Manipulationen geometrischer Objekte sowie der Anwendung der Digitalgrafik in CAD/CAM- und CAE-Systemen. Dabei werden Möglichkeiten für einen sinnvollen Einsatz der Digitalgrafik aufgezeigt. *Dr. Rainer Gebhardt*

PASCAL – Programmiertechnik für Fortgeschrittene

von P. A. Sand, McGraw-Hill Book Company GmbH Hamburg 1986, 588 Seiten

Üblicherweise verspüren Programmierer nach einer gewissen Zeit der Verwendung von PASCAL ein immer stärker werdendes Bedürfnis, das Basisanwendungsgebiet zu verlassen und Routinen zu schreiben, die anscheinend in den Bereich der Systemprogrammierung gehören oder aber als mit PASCAL nicht besonders günstig realisierbar angesehen werden. Das Buch von P. A. Sand erweist sich für derartige Fälle als eine wahre Fundgrube, und es muß anerkennend herausgestellt werden, daß die meisten Kapitel selbst für sehr erfahrene Programmierer erheblichen Erkenntniszuwachs ermöglichen.
Im einzelnen wird der Leser zuerst mit interessanten Bildschirm- und Interaktionstechniken bekannt gemacht, die für moderne Anwenderprogramme unmittelbar nutzbar sind. Weitere Schwerpunkte sind Verfahren für leistungsfähige Handhabungen von Textdateien, das Programmieren von Spielen und Simulationen (z. B. komplizierte Bewegungsabläufe auf dem Bildschirm) sowie die Darstellung von Methoden zur Erarbeitung von Programmpaketen der Klasse SuperCalc, Multiplan, ...
Zu allen Teilthemen gibt der Autor viele sehr gut aufeinander abgestimmte Programm-Module an, so daß die vorgestellten Methoden grundsätzlich am eigenen Computer geprüft bzw. eingesetzt werden können. Probleme entstehen allerdings durch die Verwendung einer Apple-PASCAL-Version, die nicht in vollem Umfang zu portablen Programmen führt. Um dieser Situation weitgehend zu begegnen, ist ein umfangreiches Kapitel aufgenommen worden, das sich mit der Portierung der Beispielprogramme auf IBM-kompatible PC unter TURBO-PASCAL beschäftigt. *Prof. Dr. Jürgen Zaremba*

FORTH-Anwendungsberichte

Die Broschüre „Applikation von FORTH" mit einem Umfang von 96 Seiten von der Sektion Technische Elektronik der WPU Rostock gibt eine gedrängte Übersicht zu einigen professionellen FORTH-Einsatzfällen und Aspekten der Systemimplementierung. Enthalten sind die Beiträge:

Woitzel/Neuthe: Basissystem comFORTH
Neuthe: Das comFORTH Stingpaket
Dünow: Grafikanwendungen in comFORTH
Drewelow: Identifikationsalgorithmen in FORTH
Pfüller: FORTH-Software für Seegang-Meßeinrichtung
Korpal: Prozeßsteuerung Brauerei
Darmüntzel: Rechnergestützte Stundenplanerstellung.

Die Broschüre (Preis etwa 10 Mark) kann bestellt werden bei: Wilhelm-Pieck-Universität Rostock, Abt. Wissenschaftspublizistik, Vogelsang 13/14, Rostock, 2500.

Software-Qualitätssicherung Aufgaben – Möglichkeiten – Lösungen

von einem Autorenkollektiv, Beuth Verlag GmbH Berlin (West) 1986. 184 S., Lit.- u. Sachwortverzeichnis

Mit dieser Veröffentlichung werden gut strukturierte Informationen zu wesentlichen Schwerpunkten des o. g. Themas vermittelt. Hauptaussagen werden getroffen zu Prinzipien der Software-Qualitätssicherung; Qualitätsmerkmale der Software; Maßnahmen, Hilfsmittel und Werkzeuge sowie Ablauf- und Aufbau-Organisation; Wirtschaftlichkeit und rechtliche Aspekte der Software-Qualitätssicherung. Dabei wird sehr deutlich, daß die Autoren sich nicht auf standardisierte Software-Merkmale stützen können. Zum Beispiel sind die fixierten Qualitätsmerkmale von Software in keiner Weise identisch mit dem einschlägigen Fachbereichsstandard (Entwurf) der DDR. Es wird auch durch die Autoren eingeräumt, daß die Zusammenhänge zwischen den einzelnen Merkmalen und die daraus abzuleitenden qualitätsorientierten Entwicklungsmaßnahmen als noch nicht vollständig geklärt angesehen werden müssen. Unbeschadet davon werden Anregungen zur Organisation der Software-Qualitätssicherung unterbreitet und eine Anzahl von Beispielen geboten, die deutlich machen, daß Software-Qualitätssicherung um so wirtschaftlicher ist, je früher sie im Entwicklungsprozeß sowohl für die konstruktiven als auch für die fehlerentdeckenden Maßnahmen praktiziert werden. *Prof. Dr. Wolfgang Schoppan*

TURBO-PASCAL

von G. Renner, Vogel-Verlag Würzburg, 2. Auflage 1986, 306 S.

TURBO-PASCAL enthält neben allen Sprachelementen von Standard-PASCAL Leistungsmerkmale, die von einem anderen Compiler bisher nicht erreicht wurden. Diese und viele andere gute Gründe führten sicherlich dazu, daß TURBO-PASCAL der PASCAL-Dialekt für Mikrocomputer schlechthin geworden ist. Das vorliegende Buch wendet sich vor allem an Anfänger und Umsteiger, die sich für eine Programmiersprache entscheiden möchten. Didaktisch gut gegliedert, wird, angefangen von der Programmierumgebung bis hin zu den Sprachelementen von TURBO-PASCAL, dem Leser eine vollständige Einführung in diese Sprache gegeben.
Zahlreiche Beispiele unterstützen dabei das theoretisch Beschriebene. Ein Fragenkomplex am Ende eines jeden Kapitels ermöglicht dem Leser eine Selbstkontrolle, inwieweit er die in dem jeweiligen Abschnitt vermittelten Kenntnisse verstanden hat.
Ein extra Kapitel ist all jenen gewidmet, die einen IBM- bzw. IBM-kompatiblen PC benutzen und die Möglichkeiten der Grafik- oder Window-Technik auch unter TURBO-PASCAL ausnutzen möchten. Auf die Besonderheiten der TURBO-PASCAL-Version, die den Arithmetik-Prozessor 8087 unterstützt sowie die für den kaufmännischen Bereich interessante Version mit BCD-Arithmetik wird ebenso eingegangen, wie auf die von der Borland Company angebotenen Softwaretools, z. B. das Turbo Access System und das Turbo Sort System, die im Source-Code ! geliefert werden und dem Anwender erlauben, z. B. leistungsfähige Datenbankprogramme zu schreiben. *Gerhard Behnke*

Anweisung	Bemerkungen	KC 85/3	KC 87	SCP-BASIC
ABS(X)	berechnet absoluten Betrag	Y=ABS(X)	Y=ABS(X)	Y=ABS(X)
AND	logische UND-Funktion	Z=X AND Y	Z=X AND Y	Z=X AND Y
ASC (X$)	ASCII-Code des ersten Zeichens	Y=ASC(X$)	Y=ASC(X$)	Y=ASC(X$)
AT	positioniert PRINT-Anweisung auf Zeile, Spalte	AT(Z, S)	AT(Z,S)	–
ATN(X)	Arcustangensberechnung im Bogenmaß	Y=ATN(X)	Y=ATN(X)	Y=ATN(X)
AUTO	automat. Zeilennumerierung	AUTO[a][,s]	AUTO[a][,s]	AUTO[a][,s]
BEEP	gibt akustisches Signal	BEEP N	BEEP N	–
BLOAD	lädt Maschinenprogramm	BLOAD	–	–
BORDER	Bildschirmrandfarbe	–	BORDER N	–
BYE	Rückkehr zum Betriebssystem	BYE	BYE	(SYSTEM)
CALL	Aufruf Maschinenprogramm Adresse dezimal	CALL adr	CALL adr	CALL adr, [parameter]
CALL*	Aufruf Maschinenprogramm Adresse hexadezimal	CALL*adr	CALL*adr	–
CDBL	wandelt einfach genaue Zahl in doppelt genaue Zahl	–	–	Y=CDBL(X)
CHAIN	Nachladen von Programmsegmenten	–	–	CHAIN [MERGE] "NAME", [Zeilen]
CHR$	liefert Zeichen mit Code von X	A$=CHR$(X)	A$=CHR$(X)	A$=CHR$(X)
CINT	erzeugt Integer-Variable	–	–	Y=CINT(X)
CIRCLE	zeichnet Kreis um X, Y mit Radius R in Farbe F	CIRCLE X, Y, R, [F]*)	–	–
CLEAR	löscht Variablenspeicher	CLEAR[Z][,E]	CLEAR [Z][,E]	CLEAR[E][,S]
CLOAD	lädt Programm von Kassette	CLOAD "NAME"	CLOAD "NAME"	–
CLOAD*	lädt Datenfeld von Kassette	CLOAD* "NAME"; FELDVar.	CLOAD* "NAME"; FELDVar.	–
CLOSE	schliesst Datei	CLOSE#n*)	–	CLOSE[#] Nummer[,...]
CLS	löscht Bildschirm	CLS	CLS	–
COLOR	stellt Farbe ein (auch als PRINT-Erweiterung)	COLOR V[,H]	–	–
COMMON	Übergabe von Informationen zwischen Programmsegmenten	–	–	COMMON Var...
CONT	setzt Programm fort	CONT	CONT	CONT
CSAVE	speichert Programm auf Kassette	CSAVE "NAME"	CSAVE "NAME"	–
COS(X)	berechnet die Cosinusfunktion	Y=COS(X)	Y=COS(X)	Y=COS(X)
CSAVE*	speichert Daten auf Kassette	CSAVE* "NAME"; FELDVar.	CSAVE "NAME"; FELDVar.	(PRINT#)
CSGN	wandelt doppelt genaue Zahl in einfach genaue Zahl	–	–	Y=CSGN(X)
CSRLIN	ermittelt Zeilennummer des Cursors	A=CSRLIN(N) *)	–	–
CVD	erzeugt doppelt genaue Zahl aus Zeichenkette	–	–	Y$=CVD(X)
CVI	erzeugt Integerzahl aus Zeichenkette	–	–	Y$=CVI(X)
CVS	erzeugt einfach genaue Zahl aus Zeichenkette	–	–	Y$=CVS(X)
DATA	vereinbart Konstantenliste	DATA C1[,C2...]	DATA C1[,C2...]	DATA C1[,C2...]
DEEK	liest zwei Speicherplätze	X=DEEK(N)	X=DEEK(N)	–

Anweisung	Bemerkungen	KC 85/3	KC 87	SCP-BASIC
DEF FN	definiert eine Funktion	DEF FNY(P) =Aus.	DEF FNY(P) =Aus.	DEF FN(P [,Pn...])=Aus.
DEFDBL	vereinbart doppelt genaue Var.	–	–	DEFDBL Var [,Var...]
DEFINT	vereinbart Integervariable	–	–	DEFINT Var [,Var...]
DEFSGN	vereinbart einfach genaue Var.	–	–	DEFSGN Var [,Var...]
DEFSTR	vereinbart Stringvariable	–	–	DEFSTR Var [,Var...]
DEFUSR	trägt Startadresse für USER-Funktion ein	–	–	DEFUSRN= Adresse
DELETE	löscht Zeilen	DELETE A,E	DELETE A,E	DELETE A,E
DIM	legt Grösse eines Variablenfeldes fest	DIM FELDNAME (I[,J...])	DIM FELDNAME (I[,J...])	DIM FELDNAME (I[,J...])
DOKE	beschreibt zwei Speicherzellen	DOKE A, D	DOKE A,D	–
EDIT	ruft Zeileneditor auf	EDITn	EDITn	EDITn
ELSE	Alternative bei IF	ELSE Anw.	ELSE Anw.	ELSE Anw.
END	beendet Programmabarbeitung	END	END	END
EOF	zeigt Dateiende an	–	–	Y=EOF (Dateinr.)
EQV	liefert Äquivalenzfunktion	–	–	Z=X EQV Y
ERASE	gibt dimensionierte Felder frei	–	–	ERASE Feldn. [,Feldn...]
ERL	enthält Zeilennummer bei ERROR	–	–	IF ERL=N THEN...
ERR	enthält Fehlercode	–	–	IF ERR=n THEN...
ERROR	simuliert Programmfehler	–	–	ERROR (Fehlercode)
EXP(X)	liefert Exponentialfunktion	Y=EXP(X)	Y=EXP(X)	Y=EXP(X)
FIELD	legt Satzstruktur fest	–	–	FIELD [#]Nr., Länge AS Var.[,...]
FILES	zeigt Dateiverzeichnis an	–	–	FILES ["Dateibez."]
FIX	liefert den ganzzahligen Teil einer einfach genauen Zahl	–	–	Y=FIX(X)
FN	ruft definierte Funktion auf	Z=FNY(P)	Z=FNY(P)	Y=FNY (P[,Pn...])
FOR	legt Programmschleifenanzahl fest	FOR Var=A TO E	FOR Var=A TO E	FOR Var=A TO E
FRE	zeigt verfügbaren Speicher an	Y=FRE(Var)	Y=FRE(Var)	Y=FRE(Var)
GET	liest Satz in den Puffer	–	–	GET[#]Dateinr.[,Satznr.]
GOSUB	ruft Unterprogramm	GOSUB n	GOSUB n	GOSUB n
GOTO	springt unbedingt	GOTO n	GOTO n	GOTO n
HEX$	wandelt Zahl in Hexadezimalzahl	–	–	Y$=HEX$(X)
IF	bedingte Sprung- oder Handlungsanweisung	IF...THEN IF...GOTOn	IF...THEN IF...GOTOn	IF...THEN IF...GOTOn
IMP	logische Implikationsfunktion	–	–	Z=Y IMP X
INK	stellt Vordergrundfarbe ein (auch als PRINT-Erweiterung)	INK N	INK N	–
INKEY$	fragt Tastatur ab	A$=INKEY$	A$=INKEY$	A$=INKEY$
INP	fragt Port ab	A=INP(I)	A=INP(I)	A=INP(I)
INPUT	wartet auf Eingabe	INPUT Var	INPUT Var	INPUT Var
INPUT#	liest Daten ein	INPUT#n Var	INPUT#n Var	INPUT#Dateinr., Var

Anweisung	Bemerkungen	KC 85/3	KC 87	SCP-BASIC
INPUT$	erwartet N Tastatureingaben	–	–	A$=INPUT$ (N)
INSTR	sucht eine Zeichenkette in einer anderen	A=INSTR (A$,B$)*)	A=INSTR (A$,B$)	A=INSTR (A$,B$)
INT	wandelt Variablenwert in Integerwert	A=INT(X)	A=INT(X)	A=INT(X)
JOYST	fragt Spielhebel ab	(vorbereitet*)	JOYST(N)	–
KEY	belegt Funktionstaste	KEY N*)	–	–
KEYLIST	listet Funktionstastenbelegung	KEYLIST*)	–	–
KILL	löscht Datei	–	–	KILL "Dateibez."
LEFT$	bildet Teilzeichenkette	A$=LEFT$ (B$,N)	A$=LEFT$ (B$,N)	A$=LEFT$ (B$,N)
LEN	liefert die Länge einer Zeichenkette	A=LEN(B$)	A=LEN(B$)	A=LEN(B$)
LET	weist Variable Wert zu (kann entfallen)	LET B=Aus.	LET B=Aus.	LET B=Aus.
LINE	zeichnet Linie	LINE XA,YA, XE, YE[,F]*)	–	–
LINES	legt Zeilenanzahl beim Listen fest	LINES N	LINES N	–
LINE INPUT	liest Zeile ein	–	–	LINE INPUT Var
LINE INPUT#	liest Zeile von Datei ein	–	–	LINE INPUT# Dateinr., Var
LIST	listet Programm aus	LIST [n]	LIST [n]	LIST [n], [m]
LIST#	listet auf Gerät aus	LIST#N "NAME"	LIST#N "NAME"	(LLIST)
LLIST	druckt Programm aus	(LIST#N)	(LIST#N)	LLIST[N], [M]
LOAD	lädt Programm	LOAD#N "NAME"	LOAD#N "NAME"	LOAD"Dateibez."[,R]
LOC	übergibt Satznummer	–	–	A=LOC (Dateinr.)
LOCATE	setzt die Cursorposition	LOCATE Z, S	–	–
LOF	zeigt Anzahl der Sätze an	–	–	A=LOF (Dateinr.)
LOG	berechnet den natürlichen Logarithmus	(LN(X))	(LN(X))	Y=LOG(X)
LN	berechnet den natürlichen Logarithmus	Y=LN(X)	Y=LN(X)	(LOG(X))
LPOS	zeigt Position des Druckkopfes	–	–	A=LPOS (Dummy)
LPRINT	gibt PRINT-Anweisung an Drucker	(PRINT#n)	(PRINT#n)	LPRINT Var...
LPRINT USING	druckt formatiert	–	–	PRINT USING "Form."; Var...
LSET	trägt Zeichenkette in Satz ein	–	–	LSET A$=X$
MERGE	mischt Programmteile	–	–	MERGE "DATEI"
MID$	erzeugt Teilzeichenkette	A$=MID$ (B$,P,N)	A$=MID$ (B$,P,N)	A$=MID$ (B$,P,N)
MKD$	wandelt doppelt genaue Zahl in 8-Byte-Zeichenkette	–	–	A$=MKD$(X)
MKI$	wandelt Integerzahl in 2-Byte-Zeichenkette	–	–	A$=MKI$(X)
MKS$	wandelt doppelt genaue Zahl in 4-Byte-Zeichenkette	–	–	A$=MKS$(X)
MOD	liefert Rest bei Division	–	–	Z=Y MOD X

Anweisung	Bemerkungen	KC 85/3	KC 87	SCP-BASIC
NAME	benennt Dateien um	–	–	NAME "neu" AS "alt"
NEW	löscht Programm	NEW	NEW	NEW
NEXT	schliesst FOR-NEXT-Schleife ab	NEXT[X[,Y..]]	NEXT[X[,Y..]]	NEXT[X[,Y..]]
NOT	Negationsfunktion	A=NOT(B)	A=NOT(B)	A=NOT(B)
NULL	legt Dummyzeichen fest	NULL N	NULL N	–
OCT$	wandelt Zahl in Oktalwert um	–	–	B$=OCT$(A)
ON GOTO	bedingte Mehrfachverzweigung	ON Var GOTO n[,m...]	ON Var GOTO n[,m...]	ON Var GOTO n[,m...]
ON ERROR	setzt fort bei Fehler	–	–	ON ERROR Reak.
ON GOSUB	bedingter Mehrfach-unterprogrammaufruf	ON Var GOSUB n[,m...]	ON Var GOSUB n[,m...]	ON Var GOSUB n[,m ...]
OPEN	eröffnet Datei	OPEN#n "NAME" *)	–	OPEN "x", #Dateinr, "Datei"
OPTION BASE	legt Indexgrenze für Felder fest	–	–	OPTION BASE N
OR	logische ODER-Funktion	Z=X OR Y	Z=X OR Y	Z=X OR Y
OUT	gibt Daten an Port	OUT P,D	OUT P,D	OUT P,D
PAPER	stellt Hintergrundfarbe ein (auch als PRINT-Erweiterung)	PAPER N	PAPER N	–
PAUSE	wartet N*0,1 s	PAUSE N	PAUSE N	–
PI	Konstante 3.14159	PI	PI	
PEEK	liest Byte von Speicherplatz	A=PEEK(I)	A=PEEK(I)	A=PEEK(I)
POKE	schreibt Byte auf Speicherplatz	POKE I,B	POKE I,B	POKE I,B
POS	gibt Cursorposit. in der Zeile	A=POS (Dummy)	A=POS (Dummy)	A=POS (Dummy)
PRESET	löscht Bildpunkt	PRESET X,Y	–	–
PRINT	gibt auf Bildschirm aus	PRINT[Erw.;] Var auch?	PRINT[Erw.;] Var auch?	PRINT[Erw.;] Var ... auch?
PRINT#	gibt auf Gerät aus	PRINT#n ...	PRINT#n ...	PRINT#n ...
PRINT USING	gibt formatiert aus	–	–	PRINT USING "Form."; Var ...
PSET	setzt Bildpunkt	PSET X, Y[,F]	–	–
PTEST	testet Bildpunkt	A=PTEST(X) *)	–	–
PUT	schreibt Satz in Datei	–	–	PUT[#] Dateinr, Satznr
RANDOMIZE	startet den Zufallsgenerator	RANDOMIZE *)	–	RANDOMIZE
READ	liest Daten aus DATA-Zeilen	READ Var ...	READ Var ...	READ Var ...
REM	kennzeichnet Kommentar	REM ... oder!	REM ... oder!	REM ... oder!
RENUMBER	numeriert Programmzeile neu	RENUMBER [a[,e[,n[,S]]]]	RENUMBER [a[,e[,n[,S]]]]	RENUMBER [n[,a[,S]]]
RESET	schliesst alle Dateien ab	–	–	RESET
RESTORE	setzt DATA-Zeiger auf Zeile	RESTORE n	RESTORE n	RESTORE n
RESUME	schliesst Fehlerbehandlung ab	–	–	RESUME Fort.
RETURN	bewirkt Rückkehr aus Unterprogramm	RETURN	RETURN	RETURN
RIGTH$	bildet Teilzeichenkette	A$=RIGHT$ (B$,N)	A$=RIGHT$ (B$,N)	A$=RIGHT$ (B$,N)
RND	berechnet Pseudozufallszahl	A=RND(S)	A=RND(S)	A=RND(S)
RSET	trägt Zeichenkette in Satz ein	–	–	RSET A$=B$
RUN	startet Programm	RUN[n]	RUN[n]	RUN[n] RUN "NAME" [,R]

Anweisung	Bemerkungen	KC 85/3	KC 87	SCP-BASIC
SAVE	rettet Programm	(CSAVE)	(CSAVE)	SAVE "NAME" [,Op.]
SGN	bestimmt Signumfunktion	A=SGN(X)	A=SGN(X)	A=SGN(X)
SIN	berechnet Sinusfunktion	Y=SIN(X)	Y=SIN(X)	Y=SIN(X)
SOUND	erzeugt Töne	SOUND T1, V1, T2, V2[,L[,D]]		
SPACE$	erzeugt Zeichenkette aus N Leerzeichen	–	–	A$=SPACE$ (N)
SPC	erzeugt N Leerstellen bei PRINT-Anweisung	SPC(N)	SPC(N)	SPC(N)
SQR	berechnet Quadratwurzel	Y=SQR(X)	Y=SQR(X)	Y=SQR(X)
STEP	bestimmt Schrittweite bei FOR-NEXT-Schleife	STEP Aus.	STEP Aus.	STEP Aus.
STOP	unterbricht Programmabarbeitung	STOP	STOP	STOP
STR$	wandelt Zahl in Zeichenkette	A$=STR$(X)	A$=STR$(X)	A$=STR$(X)
STRINGS	vervielfacht Zeichenkette	STRING $(N,B$)	STRING $(N,B$)	STRING $(N,B$)
SWAP	tauscht Variablen	–	–	SWAP A,B
SWITCH	schaltet Module	SWITCH A,S*)	–	–
SYSTEM	kehrt zum Betriebssystem zurück	(BYE)	(BYE)	SYSTEM
TAB	tabuliert bei Print-Anweisung	TAB(N)	TAB(N)	TAB(N)
TAN	berechnet Tangensfunktion	Y=TAN(X)	Y=TAN(X)	Y=TAN(X)
TROFF	schaltet Zeilennummeranz. aus	TROFF	TROFF	TROFF
TRON	schaltet Zeilennummeranz. ein	TRON	TRON	TRON
USR	ruft Funktion auf	Y=USR(X)	Y=USR(X)	Y=USR(X[, Var...])
VAL	bestimmt numerischen Wert einer Zeichenkette	A=VAL(X$)	A=VAL(X$)	A=VAL(X$)
VARPTR	bestimmt Anfangsadresse einer Variablen oder des I/O-Puffers	–	–	A=VARPTR (X) A= VARPTR#n
VGET$	liest Zeichen vom Bildschirm	A$ =VGET$ *)	–	–
VPEEK	liest Byte aus Bildwiederholspeicher	A=VPEEK(I)	–	–
VPOKE	schreibt Byte in Bildwiederholspeicher	VPOKE I,A	–	–
WAIT	wartet auf Reaktion am Port	WAIT I,J[,K]	WAIT I,J[,K]	WAIT I,J[,K]
WHILE	baut Iterationsschleife auf	–	–	WHILE Bed. Reak. WEND
WEND				
WIDTH	legt Ausgabezeilenlänge fest	WIDTH N	WIDTH N	WIDTH [LPRINT] N
WINDOW	stellt Bildschirmfenster ein	WINDOW[ZA, ZE,SA,SE]	WINDOW[ZA, ZE,SA,SE]	–
WRITE	gibt alternativ zu PRINT aus	–	–	WRITE Var, ...

ISSN 0232–2892
Mikroprozessortechnik, Berlin 2 (1988) 3
S. 65–96

Klaus-Dieter Kirves

BASIC-Sprach-übersicht

für KC 85/3, KC 87 und SCP-BASIC-Interpreter BASI


ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK
MP
Heft 4 · 1988
Mikroprozessortechnik
VEB Verlag Technik Berlin
ISSN 0232-2892
Schnelle A/D-Wandlung
Bauanleitung: Joystickmodul
Floppy-Disk-Controller U 8272 D



Info



PC-Bus

Seit kurzem rollt über die Straßen der Dagestanischen ASSR im im Osten des Nordkaukasus dieser mit 24 Personal-Computern ausgerüstete Ikarus-Bus. An den PCs werden Schüler der 9. und 10. Klassen in die Grundlagen der Computertechnik und Informatik eingeführt.
Für Schüler unterer Klassen ist die Einrichtung eines rollenden Computer-Schulraumes geplant.

Fotos: ADN-ZB/TASS



Ingenieurhochschule Berlin gegründet

Auf einem akademischen Festakt im Roten Rathaus der Hauptstadt wurde am 8. Januar 1988 die Ingenieurhochschule Berlin gegründet.
Sie geht aus der Ingenieurschule für Maschinenbau und Elektrotechnik hervor und wird vorrangig für die Berliner Kombinate Ingenieure und Ingenieurökonomen ausbilden. Zugleich erhöht sich das Forschungspotential Berlins auf den Gebieten der Elektrotechnik/Elektronik und des Maschinenbaus. Die neue Bildungsstätte bietet insbesondere jungen Facharbeitern aus der Berliner Industrie die Möglichkeit, ein Hochschulstudium aufzunehmen. Das Wirkungsfeld der Bildungsstätte soll an der Nahtstelle zwischen Wissenschaft und Produktion liegen, um aus der Anwendung der Wissenschaft in der Erzeugnis-, Verfahrens- und Technologieentwicklung höheren ökonomischen Vorteil zu ziehen, sagte Prof. Dr. h. c. Hans-Joachim Böhme, Minister für Hoch- und Fachschulwesen. An die Ingenieurhochschule wurden 11 Professoren und 10 Dozenten berufen. Rektor ist Prof. Dr. Günter Maronna.

ADN

Software fürs Studium

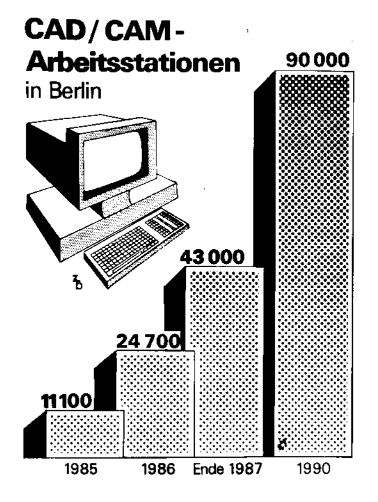
Im September 1986 begann am Institut für Film, Bild und Ton der Aufbau einer sachgebietsorientierten Leiteinrichtung „Software für Lehr- und Lernprozesse an Hoch- und Fachschulen". Eine erste Aufgabe der Leiteinrichtung besteht darin, die an allen Universitäten, Hoch- und Fachschulen bereits vorhandene mehrfachnutzbare Software zu erfassen und darüber zu informieren. Ein Katalog mit über 1100 Programmen für die Computer PC 1715, KC 85/1 und KC 85/2 wurde bereits veröffentlicht. Eine Auswahl davon steht in einer rechnergestützten Datenbank zur Verfügung, so daß Recherchen für einen breiten Nutzerkreis möglich sind. Halbjährlich wird ein Katalog über die Programmbibiliothek der Leiteinrichtung herausgegeben. Interessenten können die darin aufgeführte Software nachnutzen.
Künftig werden alle geplanten Softwareleistungen der Hoch- und Fachschulen, die einen Entwicklungsaufwand je Vorhaben von mehr als 500 Stunden pro Jahr erfordern, von der Leiteinrichtung registriert, begutachtet und bestätigt. Das vermeidet weitgehend Doppelarbeiten, fördert Kooperation und Arbeitsteilung zwischen den Hoch- und Fachschulen und erlaubt, das Potential zur Entwicklung von Softwareleistungen rationeller zu nutzen.
Die Leiteinrichtung erteilt Aufträge zur Entwicklung ausbildungsgerechter mehrfachnutzbarer Softwareleistungen, sichert die Herstellung der notwendigen Kopien und den Vertrieb an alle Hoch- und Fachschulen sowie andere Interessenten.

CAD/CAM-Arbeitsstationen

Der Bestand an CAD/CAM-Stationen hat sich von Januar 1986 bis Anfang März 1987 mehr als verdoppelt. Ende 1986 waren es 24 700. Etwa 43 Prozent sind in der zentralgeleiteten Berliner Industrie eingesetzt, rund 13 Prozent im Verkehrswesen, die übrigen Stationen vor allem im Bereich der Akademie der Wissenschaften und Staatsbank sowie bei Post und Handel.

ADN

Grafik: ADN-ZB

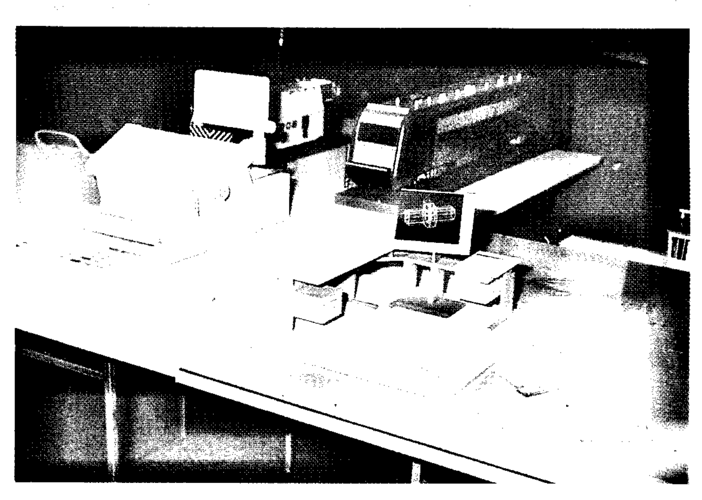


Design unterm Fernsehturm

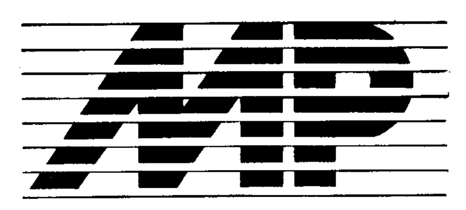
Im Januar dieses Jahres stellte die Kunsthochschule Berlin-Weißensee im Ausstellungszentrum am Fernsehturm in Berlin Arbeiten ihrer Studenten und Mitarbeiter vor. Auf dem Gebiet Design war u. a. ein Programmierarbeitsplatz (Bild 1) zu sehen. Praxispartner für die Entwicklung ist der VEB Kombinat EAW Berlin. Das Modell wurde unter Berücksichtigung der voraussichtlichen Technischen Entwicklung gestaltet. Bild 2 zeigt einen Laborschreiber und das Modell eines Konstrukteurarbeitsplatzes.

MP

Fotos (2): Paszkowsky



Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 4 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 2 87 00, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2 87 03 71); Herbert Hemke, Redakteur (Tel. 2 87 02 03); Sekretariat Tel.: 2 87 03 81

Gestaltung Christina Bauer

Titel Tatjana Stephanowitz

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

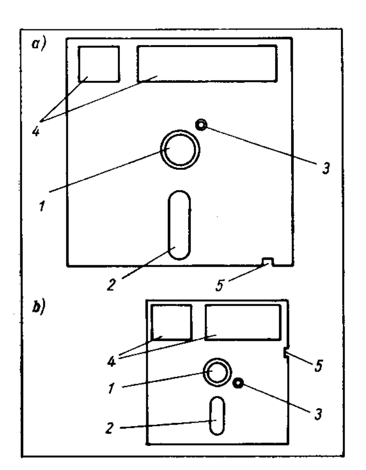
Redaktionsschluß: 16. Februar 1988

AN (EDV) 49837

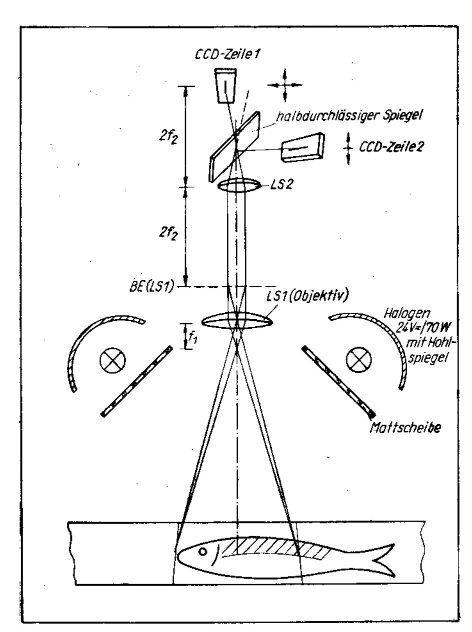
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

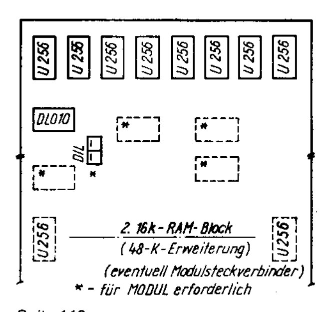
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ústřední Expedicia a Dovoz Tísku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Há Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



Seite 102



Seite 106



Seite 119

Inhalt



Dialog

Taktfrequenzumschaltung MRB Z 1013

Leider ist in dem Beitrag zur Taktfrequenzumschaltung zum Z 1013 das Bild 2 (Variante 5) fehlerhaft. Autor und Redaktion bitten um Entschuldigung.

Interessenten, die die richtige Schaltung benötigen, wenden sich an den Computerclub robotron, Gerberstr. 3, Leipzig, 7010.

Wenig beachtete Details bei der formellen Dialoggestaltung von PC-Programmen

Mit der breiten Nutzung von Personalcomputern am Arbeitsplatz werden neben Standardsoftware auch viele „maßgeschneiderte" Programme der „Marke Eigenbau" benutzt. Dabei fällt die unnötige individuelle Vielfalt bei der formellen Dialoggestaltung auf. Man ist immer gut beraten, sich die Standardsoftware als Vorbild zu nehmen. Leider ist man auch dort von allgemein anerkannten Standards noch weit entfernt. Leicht handhabbare Programme führen den Nutzer durch einfache Ja/Nein-Alternativen zu den gewünschten Programmvarianten. Dabei kann festgestellt werden, daß in der Alternative eine der Antworten meist dem sogenannten Standardfall entspricht. Das muß nicht immer der Ja-Fall sein.

Fragen von der Art:
„Datei nicht löschen (j/N)?"
sind unbedingt zu vermeiden.

Es wird empfohlen, auf eine einfache Frage die Antwort-Alternativen in der Form (j/N) oder (J/n) zur Beantwortung anzubieten, wobei der jeweils großgeschriebene Buchstabe dem Standardfall entsprechen soll, der durch Betätigen irgendeiner Taste (⟨any key⟩) – im Regelfall einfach Drücken von ⟨CR⟩ – zugewiesen wird, z. B.

„Datei FORMAL. TXT löschen (J/n)?"

Aus dem Kontext des gesamten Programms ergebe sich bei obiger Ausschrift, daß im Regelfall die Datei gelöscht werden kann.

Der Nutzer kann bei dieser Konvention durch laufendes Drücken von ⟨CR⟩ den Standardablauf durchspielen. Intern wird immer nur auf den kleingeschriebenen Anfangsbuchstaben von „... (J/n)?" geprüft, der sowohl in Klein- als auch in Groß-Schreibung und eventuell auch voll ausgeschrieben als „Nein", „no" oder „Niemals" eingegeben werden kann. Alle anderen Antworten münden in den „ELSE-Zweig".

Bei der Fallauswahl in der Art eines Menüs hat sich folgende Konvention gut bewährt:

E)dit **C**)compile **W**)ork file
S)ave **R**)un e(**X**)ecute

Dabei wird der vor der schließenden runden Klammer stehende Buchstabe in Groß- und Kleinschreibung intern als signifikant erkannt. Der Nutzer könnte dann z. B. auch eine Langform „Editieren" eingeben, wenn das programmtechnisch vorgesehen ist (BDOS(10) statt BDOS(1) unter CP/M). Eine Aufforderung, z. B.

„ENTER Arbeitslaufwerk (a..p)"

muß bei der Antwort ⟨CR⟩ entweder nochmals erfolgen oder mit einem Standardwert bzw. mit dem aus dem vorangegangenen Zyklusdurchlauf bekannten Wert automatisch erledigt werden können. Im letzteren Falle sollte der aktuelle „Standardwert" unbedingt mit angezeigt werden, so daß er mit ⟨CR⟩ quittiert oder durch einen anderen Eingabe-Wert überschrieben werden kann.

Christian Hanisch

Zum Akustikkoppler

Aufgrund der großen Nachfrage, die unsere Veröffentlichung zum Akustikkoppler in MP-Börse, Heft 8/1987, Seite 255, hervorgerufen hat, bittet uns der Entwicklungsbetrieb in Abstimmung mit dem Institut für Post- und Fernmeldewesen, Abt. Datenkommunikation, um folgende Ergänzung, deren Beachtung wir dringend empfehlen:

Nach dem Gesetz über das Post- und Fernmeldewesen GBl Teil 1 Nr. 31 vom 9. 12. 1985 ist die akustische Ankopplung genehmigungspflichtig.

Alle Akustikkoppler müssen den Empfehlungen der CCITT entsprechen, die Entwicklung und Produktion bedarf einer Herstellungsgenehmigung durch die Deutsche Post. Importierte Akustikkoppler müssen von der Deutschen Post zugelassen werden. Der im Heft 8/1987 angebotene Akustikkoppler wurde vor dem 9. 12. 1987 entwickelt und entspricht nicht den Empfehlungen der CCITT.

Eine Weiterentwicklung ist im Weimar-Werk nicht vorgesehen. Der Akustikkoppler kann deshalb ***nicht im Fernmeldenetz oder in Netzen, die mit dem Fernmeldenetz der Deutschen Post verbunden sind,*** eingesetzt werden.

Büchner
Direktor für Technik

Prüfsummen zur sicheren Programmübermittlung bei HEX-DUMPs

Auch in der MP erscheinen Programmbeispiele als HEX-DUMPs. Diese Form der Übermittlung erspart wertvollen Platz, und die Vorlagen können auch mit Drucktechnik erzeugt werden. Der große Nachteil besteht jedoch in der Eingabe in den Computer beim Leser. Jeder, der es selbst einmal praktiziert hat, kennt wohl die bange Frage vor dem Programmstart, ob alle Bytes korrekt eingegeben sind. Für die Kleincomputer KC 85/3 und auch KC 85/2 soll hier ein Vorschlag gemacht werden, der dem Leser eine einfache Möglichkeit der Überprüfung gibt.

Der Developmentmodul MO27 /1/ beinhaltet ein Prüfsummenprogramm CHSUM. Dieses Programm bildet eine Signatur ab einer Anfangsadresse in einer angegebenen Länge. Bei einem Fehler bei der Eingabe wird mit sehr hoher Wahrscheinlichkeit auch eine Änderung in der Prüfsumme auftreten.

Deshalb sollte jeder ausgedruckte HEX-DUMP gleich die zugehörige Prüfsumme enthalten. Bild 1 zeigt den DUMP des Prüfsummenprogrammes mit zugehöriger Signatur. Das Programm ist auch bei BASIC-Programmen sinnvoll, bei denen es auf Grund von POKE-, PEEK- und CALL-Anweisungen genau auf jedes Zeichen ankommt.

Bei den Softwaremodulen zum KC 85/3 werden in den EPROMs bzw. ROMs ebenfalls Prüfsummen eingesetzt. Bei 8-KByte-EPROM-Modulen erfolgt die Abspeicherung der Signaturen für alle vier Schaltkreise, einzeln gebildet, im letzten von den letzten acht Bytes. Diese vier Prüfsummenbytes werden nicht mit in die Prüfsumme des letzten EPROM eingerechnet.

Eine Überprüfung kann folgendermaßen erfolgen:

– Zuweisung des Moduls
SWITCH mm 41
(mm – Moduladresse)
– Kommandos für Prüfsummenbildung

CHSUM 4000 800
CHSUM 4800 800
CHSUM 5000 800
CHSUm 5800 7F8

Dieses Vorgehen sollte auch bei selbstprogrammierten EPROM-Modulen angewandt werden, um jederzeit den Modul überprüfen und auch bei Vervielfältigung eine einfache Kontrolle durchführen zu können.

Klaus-Dieter Kirves

Literatur

/1/ Kirves, K.-D.: Modul MO27 Development. Mikroprozessortechnik 1 (1987) 8 S. 247


```
XDISPLAY 0 40
0000 7F 7F 43 48 53 55 4D 01 __CHSUM_
0008 D5 C1 11 FF FF 7E AA 57 _______W
0010 0F 0F 0F 0F E6 0F AA 57 _______W
0018 0F 0F 0F F5 E6 1F AB 5F ________
0020 F1 F5 0F E6 F0 AB 5F F1 ________
0028 E6 E0 AA 53 5F 23 0B 78 ___S_#_x
0030 B1 20 DA EB CD 03 F0 1A _ ______
0038 CD 03 F0 2C C9 00 00 00 ___,____
XCHSUM 0 40
C3CA
```

Bild 1 HEX-DUMP des Prüfsummenprogramms mit Prüfsumme

Kommission Biotechnologie gegründet

Eine Kommission Biotechnologie beim Präsidium der Kammer der Technik ist am 27. Januar 1988 in Berlin gegründet worden. KDT-Präsident Prof. Dr. Dr. Dagmar Hülsenberg berief als Vorsitzenden Prof. Dr. sc. techn. Uwe Setzermann von der Humboldt-Universität zu Berlin.

Als Aufgaben der neuen Kommission wurden während der konstituierenden Beratung u. a. die Koordinierung von KDT-Aktivitäten zur Einführung der Biotechnologie sowie die Mitarbeit an Entwicklungs-, Überführungs- und Weiterbildungsaufgaben zu biotechnologischen Stoffwandlungsprozessen vor allem in der Nahrungsmittelproduktion, in der Pharma- und Chemieindustrie, in der Landwirtschaft sowie im Umweltschutz genannt. Die Kommission wird bei der Lösung dieser komplizierten Aufgaben die bereichs- und zweigübergreifende Gemeinschafts- und Bildungsarbeit in der KDT verstärken helfen. Der Vorrang abgestimmter Weiterbildungsarbeit in der jetzigen Entwicklungsphase der industriellen Biotechnologie wurde unterstrichen.

Die neue Kommission setzt sich aus Vertretern von biotechnologischen Produktionseinrichtungen, Ausrüstungs- und Anlagenbaubetrieben, Forschungs- und Bildungsinstitutionen sowie staatlichen Organen zusammen. Sie werden in den Fachsektionen

– Prozeßtechnik mikrobieller Stoffwandlungen
– Prozeßtechnik enzymatisch katalysierter Stoffwandlungen
– Geräte, Anlagen, Prozeßleittechnik und Labortechnik zusammenarbeiten.

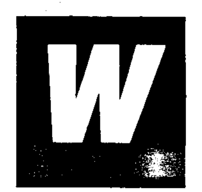
Anregungen und Mitwirkungsangebote nehmen das Präsidium der KDT, Bereich Wissenschaft und Technik/Veredlung, der Vorsitzende der Kommission sowie der stellvertretende Vorsitzende und Leiter der Arbeitsgruppe „Weiterbildung/Öffentlichkeitsarbeit", Prof. Dr. sc. techn. Hans-Jörg Raeuber, Technische Universität Dresden, entgegen.

Rb.

Das Präsidium der KDT bittet die Veränderung folgender Rufnummern ab 1. März 1988 zu beachten:

2 26 50	**Auskunft**
2 26 52 42	**Abteilung Öffentlichkeitsarbeit**
2 26 52 95	**Fachverband Bauwesen**
2 26 52 24	**Fachverband Chemische Technik**
2 26 52 31	**Fachverband Elektrotechnik**
2 26 52 37	**Fachverband Fahrzeugbau und Verkehr**
2 26 52 28	**Fachverband Holz–Papier–Polygrafie**
2 26 52 29	**Fachverband Land-, Forst- und Nahrungsgütertechnik**
2 26 52 57	**Fachverband Lebensmittelindustrie**
2 26 53 00	**Fachverband Maschinenbau**
2 26 52 33	**Fachverband Silikattechnik**
2 26 52 23	**Fachverband Textil–Bekleidung–Leder**
2 26 52 98	**Fachverband Wasser**
2 26 52 22	**Wissenschaftlich-Technische Gesellschaft für Energiewirtschaft in der KDT**
2 26 53 04	**Wissenschaftlich-Technische Gesellschaft für Meß- und Automatisierungstechnik in der KDT**

„Was lange währt wird gut", so sagt ein Sprichwort, das auch auf die Auswertung des MP-Wettbewerbs „Bestes C-Programm gesucht" zutrifft, zu dem wir in MP 7/1987 aufgerufen hatten.
Das Analysieren der Programme, einschließlich Tests, und Aufbereiten einiger Beispiele zur Veröffentlichung hat nun doch etwas mehr Zeit in Anspruch genommen.
Aber nun zur Auswertung.



Auswertung

Programmierwettbewerb „Bestes C-Programm gesucht"

Die eingeschickten Lösungen zeigten, daß alle Teilnehmer die Programmiersprache C sehr gut beherrschen. Der jüngste Teilnehmer war übrigens mit 13 Jahren der Schüler André Leopold aus Berlin. Der älteste Teilnehmer war Herr Dipl.-Ing. Lothar Janke aus Biederitz bei Magdeburg mit 36 Jahren. Das Durchschnittsalter der Teilnehmer lag erstaunlicherweise sehr niedrig, bei nur 23,9 Jahren. Es zeigt, daß vor allem die Programmiersprache C, und wahrscheinlich die Programmierung insgesamt, in starkem Maße in das Interessenfeld unserer Jugend gerückt ist. Die Mehrzahl der Einsender hatte ihre Programme auf einem Rechner getestet. Dabei waren fast alle in der DDR produzierten Rechner vertreten, wie ESER unter dem Betriebssystem OS/ES, P8000 unter dem Betriebssystem WEGA, K1630 unter den Betriebssystemen MUTOS und OS–RW, A5130 unter dem Betriebssystem MUTOS und PC 1715 unter dem Betriebssystem SCP. Diese breite Palette zeigt, daß heute C auf allen Rechnern unter fast allen Betriebssystemen verfügbar ist.
Alle eingesandten Lösungen wurden gründlich analysiert. Die jeweils drei besten Lösungen wurden ausgewählt und auf einem SKR-Rechner vom Typ SM4 unter dem Betriebssystem OS–RW von mir verifiziert und für die Veröffentlichung in diesem Beitrag aufbereitet. Die Ideen der Autoren wurden dabei nicht verändert. Bei der Aufgabe 1 wurde allerdings der Programmtext soweit gekürzt, daß das bereits veröffentlichte Rahmenprogramm (Teil VI, Bild 12) nicht noch einmal gedruckt werden mußte. Die Lösungen zur Aufgabe 2 werden in ungekürzter Fassung wiedergegeben.
Problematisch war vor allem die Auswahl der Programme für die Veröffentlichung, da fast alle Programme ihre Besonderheiten haben und deshalb vielleicht eine Veröffentlichung verdient hätten. Zum anderen haben fast alle Programme einige kleinere und größere Fehler oder Mängel, manche auch nur „Schönheitsfehler". Manchmal war gerade ein Programm mit einem interessanten Algorithmus mit kleineren Fehlern behaftet. Deshalb ließ ich mich bei der Auswahl der Programme vor allem auch davon leiten, möglichst unterschiedliche Algorithmen und Herangehensweisen in diesem Beitrag wiederzugeben.

Zur Aufgabe 1

Die Aufgabe bestand in der Entwicklung einer Ergänzung des Programms 12.4 (aus MP 6/1987) zur Suche nach Sachwörtern in einem Text, der seitenweise aufbereitet wird.
*Von allen Teilnehmern wurde die Ergänzung an der richtigen Stelle in das Programm 12.4 nach der **while**-Schleife zum Einlesen des nächsten Satzes eingepaßt. Dabei gab es zwei Herangehensweisen:*
– Einfügung des Programms zum Suchen der Sachwörter unmittelbar in den Text des Programms 12.4, was auch noch Änderungen im Programmkopf nach sich zieht, wie Vereinbarung weiterer Variablen usw.
– Programmierung als selbständige Funktion und Einfügung nur eines Funktionsaufrufes in das Programm 12.4.
Der zweite Weg ist der elegantere, weil er die Modularität unterstützt und minimale Änderungen im Programm 12.4 nach sich zieht.
*Probleme gab es bei der Vereinbarung der Sachworttabelle **swtab** mit der Endeerkennung. Die meisten Teilnehmer bestimmten die Länge der Sachworttabelle über **sizeof**-Operatoren. Andere Teilnehmer definierten zusätzlich eine leere Zeichenkette oder ein Leerzeichen als Endekennzeichen. Die einfachste Lösung ist aber die Definition eines **NULL**-Zeigers, z. B.:*
static char *swtab [] = {„Dimension", „Feld", NULL};
Bei der Realisierung des Suchalgorithmus gab es zwei grundsätzliche Herangehensweisen. Der einfachste Algorithmus ist, wenn man ab jedem Zeichen der Textzeile testet, ob ein Sachwort enthalten ist. Bei vielen Sachwörtern kommt es zu sehr langen Suchzeiten. Kritisch ist auch, daß bei kleingeschriebenen Sachwörtern es zu einem positiven Resultat kommt, wenn das Wort eine Teilzeichenkette eines anderen Wortes ist, z. B. „gehen" ist auch in „eingehende" und „Vergehen" enthalten. Da Sachwörter aber in der Regel Substantive sind, ist das Problem nicht überzubewerten.
Eine Verbesserung des Zeitverhaltens kann erreicht werden, wenn nur bei Buchstaben der Textzeile Sachwörter gesucht werden. Der exakte Algorithmus geht aber von der Erkennung der Wörter aus, wenn auf Wortlücken analysiert wird, wobei die Interpunktion zu beachten ist.
*Herr Gundlach (28) aus Potsdam zeigte eine Funktion **find()**, die auf Buchstaben in der Textzeile analysiert und das Wort in einen Puffer **puf** kopiert (Bild 1). Für die Analyse auf Buchstaben verwendete er die Standardfunktion **isalpha()**. Da nicht unter allen C-Systemen die Funktion verfügbar ist, sei sie in Bild 2 dargestellt. Für den Vergleich entwickelte er die Funktion **cmpstr()** und benutzte für die Bestimmung der Zeichenkettenlänge die Standardfunktion **strlen()**. Für Kontrollzwecke druckt er neben dem Sachwort und der Seitennummer noch das im Text gefundene Sachwort aus.*
*Herr Stange (18) aus Strausberg entwickelte eine sehr einfache Funktion **suche()** (Bild 3). Er bestimmt die Länge der Sachworttabelle über **sizeof**-Operatoren. Die Suche der Sachwörter wird über eine geschickte Anwendung von Zeigern realisiert. Wegen der Einfachheit des Algorithmus ist es eine sehr gelungene Lösung.*
Herr Knobloch (28) aus Reez bei Rostock realisierte ebenfalls eine sehr einfache Lösung und ergänzte noch die Ausgabe der Zeilennummer, in der das Sachwort gefunden wurde. Er integrierte seine Funktion in das Hauptprogramm. Um nicht das ganze Programm drucken zu müssen, wurde sein Algorithmus in Form der in Bild 4 enthaltenen Funktion wiedergegeben.
Herr Knobloch hat außerdem noch zwei elegante Funktionen zur Konvertierung Römisch-Arabisch unter Anwendung einer Struktur für die Konvertierungstabelle entwickelt.
*Einen völlig anders gearteten Algorithmus hat Herr Oehmichen aus Dresden vorgeschlagen (Bild 5). Er verwendete die ältere Standardfunktion **index()** zum Suchen, ob der erste Buchstabe eines Sachwortes in der Textzeile enthalten ist. Wenn der Buchstabe gefunden wurde, sollten die Textzeile und das Sachwort auf Gleichheit unter Anwendung der Standardfunktion **streq()** verglichen werden. Dieser Vorschlag ist originell und wurde deshalb zur Veröffentlichung aufbereitet. Dabei mußten folgende Änderungen vorgenommen werden:*
*– Die Funktion **index()** steht in moderneren C-Systemen in der Regel nicht mehr zur Verfügung. Es sollte dafür die Funktion **strchr()** verwendet werden: **position = strchr (string, character)***
*Sie gibt die Adresse zurück, wo das Zeichen **character** in der Zeichenkette **string** gefunden wurde.*
*– Die Funktion **streq()** vergleicht auf Gleichheit der Zeichenketten, d. h. sie müssen auch gleich lang sein! Die Funktion **strncmp()** dagegen vergleicht nur die ersten **n** Zeichen der zwei Zeichenketten. Die Länge **n** wird zweckmäßigerweise mit **strlen()** vom zu suchenden Sachwort bestimmt.*

Man sieht an dem Beispiel, daß es sinnvoll ist, sich mit den umfangreichen C-Bibliotheken zu beschäftigen, da sie vielfältige Funktionen (C-Tools) zur Rationalisierung der Programmierung enthalten. Aus Platzgründen konnte im MP-Kurs „Programmierung in C" darauf leider nicht eingegangen werden.

Zur Aufgabe 2

*Die Aufgabe 2 bestand in der Auswertung von Meßergebnissen eines physikalischen Versuchs, wobei die Meßergebnisse in Gleitpunktdarstellung im ASCII-Kode vorliegen. Aus bestimmten Gründen wurden immer zwei Werte in einem Satz im Datenfile **TEST.DAT** gespeichert. Da keine Dimensionen angegeben wurden, ist anzunehmen,*

```
#include <stdio.h>
static char *swtab[]={"Feld", "Dimension", "\Ø"};
find(f1, sn)
char f1[], sn[];            /* Zeile + Seitennummer */
{
char puf[1ØØ], **z, *zg;
int merk, i, j;
merk = Ø;
/* Abarbeiten einer uebergebenen Zeile */
for(i=Ø; f1[i] != '\Ø'; i++)
        {
        if(!merk)       zg = puf;        /* Zeiger auf Wortmerker */
        if(isalpha(f1[i]))       {
                *zg++ = f1[i];
                merk = 1;
                }
        else    {
                *zg = '\Ø';      /* Ende Zeichenkette */
                merk = Ø;
                z = swtab;
                for(j=Ø; *swtab[j] != '\Ø'; j++, z++)
                    if(cmpstr(puf, *z))
                        printf("%s\t\t%s\t\t(%s)\n", *z, sn, puf);
                }
        }
}

cmpstr(s1, s2)
char *s1, *s2;
{
int n;
n = strlen(s2);
while(*s1++ == *s2++ && n)      n--;
return(n>Ø ? Ø : 1);
}
```

Bild 1 Funktion find() nach Gundlach

```
isalpha(c)
char c;
{
return((c>='A' && c<='Z') || (c>='a' && c<='z') ? 1 : Ø );
}
```

Bild 2 Funktion isalpha()

▼ Bild 3 Funktion suche() nach Stange

```
#include <stdio.h>
static char *swtab[]= {"Dimension", "Feld"};
suche(sw,s)
/* Gibt alle Sachwoerter von swtab aus, die in sw
   in den spezifizierten Zeichen uebereinstimmen */
char *sw,  /* Zeiger auf 1. Zeichen */
     s[];  /* Seitennummer */
{
/* Anzahl der Sachwoerter in swtab */
static int w={ sizeof swtab/sizeof(char *) };
char *c, *c1, *c2;
int i;

for (c=sw;*c!='\Ø';c++)
   {
  for (i=Ø; i<w; i++)  /* Vergleich mit allen Sachwoertern */
     {
        c1=c; c2=swtab[i];       /* Zeiger auf 1. Zeichen */
        for(;*c1==*c2 && *c2!='\Ø';c1++) c2++; /* Vergleich */
        /* Ausgabe des Sachwortes mit Seitennummer wenn
         gefunden */
        if(*c2=='\Ø') printf("Seite:%s  Wort:%s\n",s,swtab[i]);
     }
   }
}
```

```
#include <stdio.h>
static char *swtab[]= {"Dimension", "Feld", ""};
find(feld,sn)
char feld[], sn[];
{
static int n=Ø;
for (c=feld; *c; c++)
   {
     /* Test, ob Wort in Tabelle */
     for (ztab=swtab; **ztab; ztab++)
        {
          if (**ztab==*c)
            {  /* erster Buchstabe gleich */
               ztxt = c;
               for(ztab2 = *ztab; *ztab && *ztab2++ == *ztxt++;);
               if (*(ztab2 - 1)) continue;
               printf("\"%s\"\tauf Seite %s, Zeile %d \tgefunden\n",
                          *ztab, sn, n+1);
               break;
            }
          }
   }
n++;
}
```

Bild 4 Funktion find() nach Knobloch

```
#include <stdio.h>
static char *swtab[]={"Feld", "Dimension"};
find(feld, sn)
char feld[], sn[];
{
char *posit;
int i;
/* Sachwoerter suchen */
for(i=Ø; i<sizeof swtab/sizeof (char *); i++)
        for(posit=feld;(posit=strchr(posit,*swtab[i]))!=NULL;posit++)
                if(strncmp(swtab[i],posit,strlen(swtab[i]))==NULL)
                        printf("%-2Øs\t%1Øs\n",swtab[i],sn);
}
```

```
#include <stdio.h>
#include <math.h>
main()
{
/* DEFINITIONEN */
static char *text[]={
     "\n            MINIMUM",
       "            MAXIMUM",
       "         MITTEWLWERT",
       "Standardabweichung"
};
FILE *fp1,*fp2;
float wert,summe,erg[4];
short anz;
/* ANFANGSBEDINGUNGEN */
anz=Ø;summe=Ø;
if((fp1= fopen("test.dat","r"))==NULL)
        error ("test.dat can't be opened");
if((fp2= fopen("/dev/lp1","w"))==NULL)
        error ("lp1 can't be opened");
/* MITTELWERTBERECHNUNG;MIN.,MAX. BESTIMMEN */
while (fscanf (fp1,"%f",&wert) != EOF)
        {
        if (anz++==NULL)
                {erg[Ø]=wert; erg[1]=wert;}
        if (wert<erg[Ø])
                erg[Ø]=wert;
        if (wert>erg[1])
                erg[1]=wert;
        summe+=wert;
        }
erg[2]=summe/anz;
/* MESSWERTE DRUCKEN;STANDARDABWEICHUNG BERECHNEN */
rewind(fp1); summe=Ø; anz=Ø;
fprintf(fp2,"\n\t\tM E S S W E R T E\n\n");
while (fscanf(fp1,"%f",&wert)!=EOF)
        {
        summe+=(wert-erg[2])*(wert-erg[2]);
        fprintf(fp2,"%12.3f",wert);
        if(++anz%5==NULL)
                fprintf(fp2,"\n");
        }
erg[3]=sqrt((double) summe/(anz-1));
/* ERGEBNISSE DRUCKEN */
for (anz=Ø;anz<4;anz++)
        fprintf(fp2,"\n%s: %12.3f",text[anz],erg[anz]);
fprintf(fp2,"\n");
/* PROGRAMMABSCHLUSS */
fclose (fp1);fclose(fp2);
}
/*FEHLERAUSGABE*/
error (cfp)
char *cfp;
{
printf("%s \n",cfp);
}
```

Bild 6 Lösung zur Aufgabe 2 von Janke

```
#include <stdio.h>
#include <math.h>
#define ANZAHL 2ØØØ
main()
{
/* Vereinbarungen */
int i,j,n;
float s,min,max,qusum,sum,mittel;
float mw[ANZAHL];           /* Messwertfeld */
char zeile[3Ø];             /* eingelesene ASCII-Zeichenzeile */
FILE *fptest,*fpdr;         /* Textfile,Drucker */
/* Initialisierungen */
n= -2;                      /* Anzahl Messwerte */
s=Ø;                        /* Standardabweichung */
min=max=Ø;                  /* kleinster, groesster Wert */
sum=mittel=Ø;               /* Summe, Mittelwert */
qusum=Ø;                    /* Quadratsumme (Messwert-Mittelwert) */
/* Eroeffnung der Files */
fptest=fopen("TEST.DAT","r");
fpdr=fopen("LP:","w");
/* Einlesen Datensaetze */
while (fgets(zeile,3Ø,fptest)&&(n<ANZAHL))
        {
        n += 2;       /* Ein Datensatz = 2 Messwerte eingelesen */
/* Umwandlung ASCII-Zeile in 2 Zahlen */
        sscanf (zeile,"%f%f",&mw[n],&mw[n+1]);
/* min., max. Messwert */
        for (j=n;j<n+2;j++)
                if (mw[j]<min) min=mw[j];
                else
                          if (mw[j]>max) max=mw[j];
/* Summe der Messwerte */
        sum +=mw[n]+mw[n+1];
        }
/* Mittelwert */
mittel=sum/n;
/* Standardabweichung */
for (j=Ø;j<n;j++)
        qusum +=(mw[j]-mittel)*(mw[j]-mittel);
s=sqrt((double)(1./(n-1)*qusum));
/* Ausgabe der Messwerte in 5 Spalten zu 12 Stellen */
for (j=Ø;j<n;)
        {
        putc ('\n',fpdr);                 /* Zeilenwechsel */
        for (i=Ø;(i<5)&&(j<n);i++)        /* 5 Spalten */
                fprintf(fpdr,"%12.3f",mw[j++]);
        }
/* Ausgabe der Ergebnisse */
fprintf (fpdr,"\nmin. Wert = %.3f",min);
fprintf (fpdr,"\nmax. Wert = %.3f",max);
fprintf (fpdr,"\nMittelwert = %.3f",mittel);
fprintf (fpdr,"\nStandardabw. = %.3f",s);
/* Schliessen der Files */
fclose(fptest);
fclose(fpdr);
}
```

Bild 7 Lösung zur Aufgabe 2 von Walther

◀ Bild 5 Funktion find() nach Oehmichen

```c
#include <stdio.h>
#include <math.h>
main()
{
FILE *fdat, *fpr;
static float mw[2ØØØ], mitt=Ø.Ø, mqa=Ø.Ø; /* Messwerte */
static int idx=Ø, maxi=Ø, mini=Ø;          /* Indizes */
int n, i;

if((fdat=fopen("TEST.DAT","r"))==NULL)     /* Fileeroeffnung */
   error ("Kann Messwertfile nicht oeffnen \n");
if((fpr=fopen("/stdpr","w"))==NULL)
   error ("Drucker nicht verfuegbar \n");

while (fscanf(fdat,"%f %f",&mw[idx++],&mw[idx++]) != EOF);
                                           /* Messwertuebernahme */
if(!(n=idx-2))
        error ("Messwertfile leer \n");

for(idx=Ø;idx<n; idx++)
        {
        mitt += mw[idx];
        mqa  += mw[idx] * mw[idx];
        if (mw[idx] > mw[maxi])
                maxi = idx;
        else if (mw[idx] < mw[mini])
                mini = idx;
        }

mqa = (float)sqrt((double)((mqa-mitt*mitt/n)/(n-1)));
mitt /= n;
fprintf (fpr,"Messergebnisse :\n");
for (idx=Ø; idx<n; idx+=i)
        {
        for (i=Ø; i<5 && idx+i<n; i++)
                fprintf (fpr,"%12.3g",mw[idx+i]);
        fprintf (fpr,"\n");
        }
fprintf (fpr," Anzahl der Messwerte: %d\
 \n Minimum: %.3g \n Maximum: %.3g\
 \n Mittelwert: %.3g\
 \n Mittlere quadratische Abweichung: %.3g \n",
 n,mw[mini],mw[maxi],mitt,mqa);
fclose (fpr);
fclose (fdat);
}

error(fm)                             /* error - Funktion */
char *fm;
{
fprintf(stderr,fm);
exit(1);
}
```

Bild 8 Lösung zur Aufgabe 2 von Ossenkopf

daß alle Werte die gleiche Dimension haben.

Das erste Problem bei der Programmierung war die Umwandlung der Gleitpunktzahlen in das interne **float**-*Format. Hierzu kann die Funktion* **fscanf()** *genutzt werden.*

*Problematisch war die End-of-File-Erkennung (***EOF***), da im Teil 5 des MP-Kurses „Programmierung in C" bei der Beschreibung der Funktion* **fscanf()** *aus Platzgründen keine Aussage getroffen wurde.*

fscanf() *übergibt einen* **int**-*Wert, der die Anzahl der fehlerfrei umgewandelten Parameter angibt. Wenn das Fileende erkannt wurde, übergibt* **fscanf()** *(und* **scanf()***) den Wert −1 (***EOF***). Wenn man diese Möglichkeit nicht kennt, kann man den Datensatz mit* **fgets()** *einlesen und mit* **sscanf()** *umkonvertieren. In der Praxis wird von dieser Variante häufig Gebrauch gemacht.*

Auch kann die Funktion **fscanf()** *nur zum Konvertieren eines Wertes genutzt werden, auch wenn im Datensatz mehrere Werte gespeichert sind. Sobald ein Datensatz verarbeitet ist, liest* **fscanf()** *bei Erfordernis den nächsten Datensatz ein, d. h. die Formatbeschreibung in* **fscanf** *ist nicht an die Datensatzstruktur gebunden. Ein nächstes Problem war die Ermittlung des minimalen und maximalen Wertes. Kritisch ist hier der Anfangswert der Variablen für das Minimum bzw. Maximum. Der Anfangswert Null ist ungeeignet. Deshalb wurde in manchen Lösungen für die Maximum-Variable der „kleinste" Wert und für die Minimum-Variable der „größte" Wert als Anfangswert gesetzt. Was aber der kleinste oder größte Wert ist, darüber gingen die Auffassungen stark auseinander. Das Spektrum reichte von −1E30 bzw. +1E30 bis zu −1E51 bzw. +1E51. Tatsächlich ist der kleinste und größte darstellbare Wert compiler- bzw. rechnerabhängig. Eine solche Programmierung führt zwangsläufig zu einem nicht portablen oder inkorrekten Programm. Gut beraten war man, wenn man sich aus der Menge der Meßwerte einen beliebigen Wert, in der Regel den ersten, als Anfangswert herausnahm. Bei der Berechnung des Mittelwertes gab es keine Probleme, dafür aber bei der mittleren quadratischen Abweichung, die laut „Kleiner Enzyklopädie Mathematik" auch als Standardabweichung* **s** *bezeichnet wird und sich aus der Wurzel der Streuung oder Varianz* $\mathbf{s^2}$ *ergibt. Die Varianz* $\mathbf{s^2}$ *wird wie folgt berechnet:*

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_1 - \overline{x})^2 \qquad (1)$$

Die Anwendung dieser Formel setzt erst die Berechnung des Mittelwertes $\overline{x}$ *voraus und dann in einem zweiten Durchlauf die Berechnung der Varianz* $\mathbf{s^2}$*. Hierbei bieten sich zwei Lösungen an:*

- *Das Datenfile wird zweimal eingelesen, was natürlich viel Zeit kostet.*
- *Die Meßergebnisse werden in einer Tabelle im Hauptspeicher zwischengespeichert, wodurch das zweite Einlesen entfällt. Zur Erleichterung der Programmierung dieser Variante war in der Aufgabenstellung gesagt, daß die Anzahl der Meßwerte kleiner 2000 ist.*

Ein Vereinfachung der Programmierung ergibt sich aber, wenn man den Ausdruck unter dem Summenzeichen nach der binomischen Formel auflöst:

$$s^2 = \frac{1}{n=1} \left(\sum_{i=1}^{n} x_i^2 - 2\overline{x} \sum_{i=1}^{n} x_i + \overline{x}^2 n \right) \qquad (2)$$

Durch Anwendung der Formel (2) kann man in einem Durchlauf der Werte sowohl den Mittelwert als auch die Varianz berechnen, indem man die Summe der Meßwerte, die Summe der Quadrate der Meßwerte und die Anzahl der Meßwerte berechnet.

In vielen Einsendungen wurde auch die Formel (2) durch Ersetzen des Mittelwertes $\overline{x}$ *weiter vereinfacht:*

$$s^2 = \frac{1}{n-1} \left(\sum_{i=1}^{n} x_i^2 - \frac{1}{n} \left(\sum_{i=1}^{n} x_i \right)^2 \right) \qquad (3)$$

Die Ausgabe der Meßwerte in 5 Spalten und der Auswertungsergebnisse bereitete im allgemeinen keine Probleme.

Herr Janke (36) aus Biederitz bei Magdeburg löste diese Aufgabe durch zweimaliges Einlesen der Meßdatenfiles auf einem K1600 unter MUTOS (Bild 6).

Herr Walther (24), Student aus Karl-Marx-Stadt, zeigt in seiner Lösung das Ablegen der Meßergebnisse in einer Tabelle **mw** *(Bild 7). Das Programm war nicht getestet und lief nach dem Eingeben und einer Korrektur auf unserem Rechner. Bei der Berechnung des Radikanten für die Funktion* **sqrt()** *mußte nach der Konstanten l ein Punkt eingefügt werden. Ohne den Punkt wurde eine Integerdivision ausgeführt und der Radikant war gleich Null. Die Typwandlung durch die Angabe (***double***) erfolgt erst nach der Ausdrucksberechnung. Natürlich müßte noch eine Änderung des Setzens der Anfangswerte für* **min** *und* **max** *eingearbeitet werden, damit der minimale und maximale Wert richtig berechnet werden können. Außerdem müßte noch das Testen der Filepointer ergänzt werden, damit das Programm zuverlässig und sicher arbeiten kann, denn die Wahrscheinlichkeit, daß beim* **fopen()** *ein Fehler auftritt, ist doch sehr hoch.*

Herzlichen Glückwunsch Herr Walther!

Herr Ossenkopf (22), Physikstudent in Jena, wendete sofort Formel (3) an und vereinfachte sich die Programmierung wesentlich (Bild 8). Er testete übrigens seine Programme auf einem ESER-Rechner unter dem Betriebssystem OS/ES.

Auch viele andere Lösungen, wie z. B. von Herrn Stange (18) aus Strausberg, der Formel (2) verwendete, wären einer Veröffentlichung an dieser Stelle wert gewesen.

Abschließend sei allen Einsendern für ihre Mühe gedankt. Leider konnten nicht alle eingesandten Lösungen veröffentlicht werden.

Unter Ausschluß des Rechtsweges wurden aus den guten und sehr guten Lösungen die Gewinner des Buches „UNIX und C – Ein Anwenderhandbuch" ermittelt.

Herzlichen Glückwunsch allen Gewinnern!

Doz. Dr. sc. techn. Thomas Horn

Die Namen der Gewinner lauten:

Doetzkies, Uwe, *Dresden*
Grunert, Jürgen, *Berlin*
Gundlach, Peter, *Potsdam*
Janke, Lothar, *Biederitz*
Knobloch, Manfred, *Reez*
Oehmichen, Frank, *Weixdorf*
Ossenkopf, Volker, *Jena*
Stange, René, *Strausberg*
Thamm, Oliver, *Leipzig*
Walter, Carsten, *Karl-Marx-Stadt*

Der Floppy-Disk-Controller U 8272 D und sein Einsatz (Teil 1)

Dr. Eberhard Böhl
VEB Forschungszentrum Mikroelektronik Dresden

Mit der steigenden Produktion von Büro- und Personalcomputern wächst auch der Bedarf an Folienspeicherlaufwerken – wird doch die Leistungsfähigkeit eines solchen Rechners durch diesen externen Speicher wesentlich erhöht. Auch mit der Erhöhung des Integrationsgrades von Speicherschaltkreisen werden die Folienspeicher mit ihrem niedrigen Bitpreis, dem problemlosen Datenerhalt bei Spannungsabschaltung und dem einfachen off-line-Datentransfer weiterhin große Bedeutung behalten, zumal die Entwicklung auch bei Disketten und den Laufwerken in Richtung immer größerer Speicherkapazität bei verkleinertem Volumen geht.
Der Floppy-Disk-Controller U 8272 D ist ein hochintegrierter Schaltkreis zur Vereinfachung der Ansteuerung von Folienspeicher-(Disketten-)Laufwerken. Er wurde im VEB Forschungszentrum Mikroelektronik Dresden im Auftrag des VEB Kombinat Robotron entwickelt. Produzent ist der VEB Mikroelektronik „Karl Marx" Erfurt. Die Produktion erfolgt in den zwei Typen U 8272 D04 (4 MHz für Minidisketten) und U 8272 D08 (4 MHz und 8 MHz, vorzugsweise für Normaldisketten).
Nach einem kurzen Überblick über die Funktion und die technischen Daten der Diskettenlaufwerke wird in diesem Beitrag insbesondere auf das vom Floppy-Disk-Controller (FDC) U 8272 D realisierte Aufzeichnungsformat und die Funktion eingegangen.

Die Diskette

Auf einer flexiblen kreisrunden Kunststoffolie wird eine dünne Schicht magnetischen Materials aufgebracht. Zum Schutz der Beschichtung dieser Diskette (floppy disk, Speicherfolie) vor mechanischer Zerstörung und Verunreinigung wird die Folie von einer festen Hülle mit einer speziellen Auskleidung umschlossen. Die filzartige Auskleidung sorgt für eine Verringerung der Reibung zwischen Diskette und Hülle bei Ableitung der elektrostatischen Aufladungen und Aufnahme von Staubteilchen und anderen kleinen Fremdkörpern. In der Hülle befinden sich Öffnungen für den Antriebsmechanismus (großes Loch in der Mitte), für das Indexloch der Diskette (kleines Loch, kennzeichnet Spuranfang) sowie für den Zugriff des Lese-Schreibkopfes (ovales Loch senkrecht zu allen Spuren (Bild 1). Das Indexloch wird durch eine Lichtschranke im Laufwerk erkannt und ist in der Regel asymmetrisch angebracht, damit die Seiten nicht vertrauscht werden.

Dr.-Ing. Eberhard Böhl (38) studierte von 1967 bis 1972 an der Technischen Hochschule Karl-Marx-Stadt (jetzt Technische Universität), Sektion Informationstechnik. Er promovierte 1976 mit einer Arbeit auf dem Gebiet der dynamischen Fehlererscheinungen in digitalen Schaltungen.
Seit 1978 ist Dr. Böhl am Institut für Mikroelektronik Dresden (jetzt Forschungszentrum Mikroelektronik Dresden) als Mitarbeiter bzw. Gruppenleiter im Entwurf von peripheren Rechnerschaltkreisen beschäftigt.

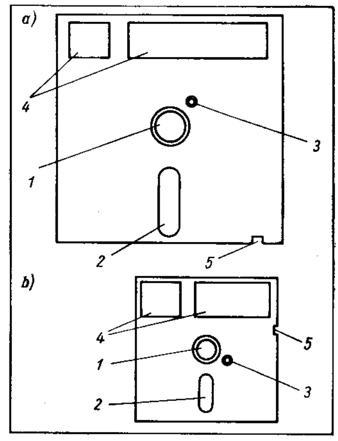


Bild 1 Aufbau von Normal- (a) und Minidisketten (b)
1 – Antriebsloch, 2 – Ausschnitt zum Kopfzugriff, 3 – Indexloch, 4 – Beschriftungsetiketten, 5 – Schreibschutzkerbe

Das Laufwerk

Beim Lesen oder Schreiben von Daten wird der Kopf im Laufwerk unmittelbar in Kontakt mit der Diskette gebracht. Da die dünne magnetische Schicht nur etwa eine Million Umdrehungen mit Kopfkontakt verträgt (bei 360 min^{-1} ca. 50 Betriebsstunden), wird der Kopf nur zum Lesen oder Schreiben geladen – die Steuerung des Kopfes wird durch den U8272D entsprechend den vorliegenden Befehlen vorgenommen. Je nach Laufwerktyp ist im U 8272 D eine entsprechende Wartezeit nach dem Auslösen der Entladung oder der Ladung des Kopfes programmierbar, damit die nachfolgenden Operationen sicher ablaufen. Der Kopf wird im entladenen Zustand in radialer Richtung durch einen Schrittmotor von Spur zu Spur bewegt. Die Impulsfolgefrequenz ist im U 8272 D programmierbar und damit an verschiedene Laufwerktypen anpaßbar. Nach der Diskettengröße unterscheidet man Laufwerke im 8-Zoll- (Normaldiskette), 5,25-Zoll- (Minidiskette) und 3- bzw. 3,5-Zoll-Format (Mikrodiskette).
In der DDR sind in der Regel nur Normal- und Minidiskettenlaufwerke im Einsatz. In den 70er Jahren dominierte zunächst die Normaldiskette mit 256 KByte Speicherkapazität auf 77 Spuren. Bei einer Umdrehungsgeschwindigkeit von 360 min^{-1} und einer step-rate-time (Impulsfolge für den Schrittmotor) von ca. 10 ms beträgt die mittlere Zugriffszeit ca. 250 ms. Durch die Anwendung der doppelten Aufzeichnungsdichte und zweiseitige Beschreibung der Disketten (auf jeder Seite ein Lese-/Schreibkopf) konnte die formatisierte Speicherkapazität schon bald auf 1 MByte erhöht werden. Die Minidisketten erreichten Ende der 70er Jahre bei einfacher Speicherdichte und einseitiger Beschreibung ca. 70 KByte formatisierte Speicherkapazität. Bei der Geschwindigkeit von 300 min^{-1} und der step-rate-time von ca. 40 ms konnten mittlere Zugriffszeiten von ca. 450 ms realisiert werden.
Schon bald wurde die Umdrehungsgeschwindigkeit auch auf 360 min^{-1} erhöht (Realisierung des 8-Zoll-Formates mit 77 Spuren) und die doppelte Aufzeichnungsdichte eingesetzt. Mit dem Einsatz von doppelseitigen Laufwerken und der Erhöhung der Spuranzahl auf 80 wird von den Minidisketten heute schon nahezu die Speicherkapazität von den Normaldisketten erreicht, und die Entwicklung geht weiter. Aus dieser Tendenz ist abzuleiten, daß der Einsatz von Normaldisketten in der Zukunft international weiter zurückgehen wird, während der Einsatz von Mini- und Mikrodisketten (bei denen auch größere Spurdichten möglich sind) erheblich steigen wird /1/.

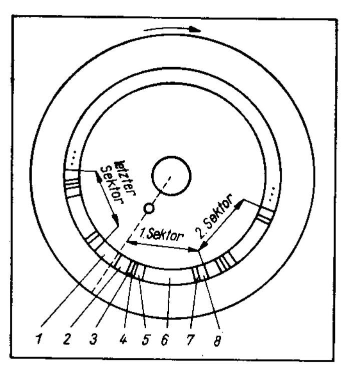


Bild 2 Schematische Darstellung des Aufzeichnungsformates einer Spur auf der Diskette
1 – Vorindexlücke, Synchronfeld, Indexmarke, Nachindexlücke;
2 – Synchronfeld, Adreßmarke, Identifikationsfeld;
3 – CRC;
4 – Adreßlücke;
5 – Synchronfeld, Datenmarke;
6 – Daten;
7 – CRC;
8 – Datenlücke

Sektorierung der Disketten

Die Disketten werden in jeder Spur in Sektoren eingeteilt, in denen eine vorgegebene Anzahl von Bytes gespeichert werden kann. Bei der *Hardsektorierung* wird jeder Sektor durch ein zusätzliches Loch in der Diskette markiert; ein an die Lichtschranke angeschlossener Zähler, der durch den Impuls vom Indexloch jeweils zurückgesetzt wird, gibt die Nummer des laufenden Sektors an. Bei der *Softsektorierung* werden die Sektoren durch je einen Vorspann mit Marken und der Angabe der Sektornummer und anderen Informationen gekennzeichnet (Bild 2). Durch Lesen des Vorspanns nach dem Kopfladen kann deshalb sofort entschieden werden, ob der nächste Sektor der zu bearbeitende ist, ohne daß vorher das Indexloch abgewartet wird. Zu beachten ist, daß durch die Formatierung der Spur mit Synchronfeldern, Marken, Identifikationsfeldern und Lücken die theoretisch mögliche Speicherkapazität für die eigentlichen Daten geringer wird.
Der U 8272 D unterstützt nur softsektorierte Aufzeichnungsmethoden mit wählbarer Sektorgröße von 128, 256, 512, 1024, 2048 oder 4096 Byte/Sektor bei einfacher Aufzeichnungsdichte bzw. 256, 512, 1024, 2048, 4096 oder 8192 Byte/Sektor bei doppelter Aufzeichnungsdichte. Es wird vorgesehen, noch in diesem Jahr zusätzlich das Format

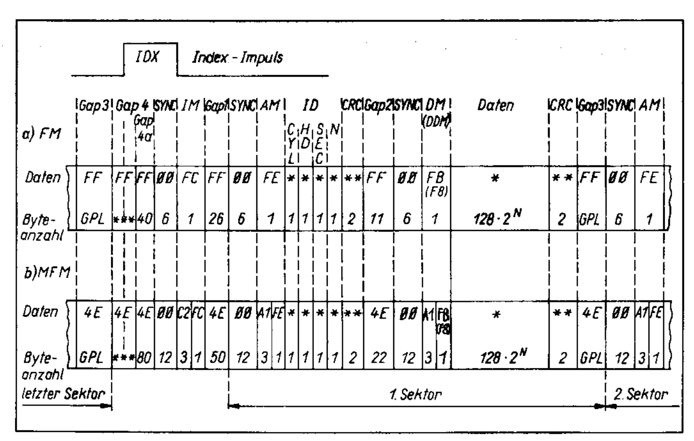


Bild 3 Aufzeichnungsformate des U 8272 für FM (a) und MFM (b)
Gap 1: Nachindexlücke Gap 2: Adreßlücke Gap 3: Datenlücke **Gap 4:** Vorindexlücke Gap 4a: konstanter Teil der Vorindexlücke
SYNC: Synchronfeld
IM: Indexmarke
AM: Adreßmarke
ID: Identifikationsfeld
CYL: Spur- bzw. Zylindernummer
HD: Kopfadresse
SEC: Sektornummer
N: Kennzeichnung der Anzahl Bytes/Sektor
CRC: Prüfbytes (cyclic redundancy code, zyklischer Redundanzcode)
DM: Datenmarke
DDM: ignorierte Datenmarke
GPL: Lückenlänge (programmierbar);
* ≙ variable Datenwerte, ** ≙ Werte entsprechend Bildungsgesetz des verwendeten Prüfpolynoms, *** ≙ Byteanzahl zur Ausfüllung der Spur vom Ende des letzten Sektors bis zum Indexloch

Tafel 1 Beziehungen zwischen den Sektorgrößen

8"-Standard-Floppy						5 1/4"-Mini-Floppy				
Sektorgröße in Byte/Sektor	N	SC	GPL[1]	GPL[2]	Bemerkungen	Sektorgröße in Byte/Sektor	N	SC	GPL[1]	GPL[2]
FM-Modus										
128	00	1A	07	1B	IBM-Disk.1	128	00	12	07	09
256	01	0F	0E	2A	IBM-Disk.2	128	00	10	10	19
512	02	08	1B	3A		256	01	08	18	30
1024	03	04	47	8A		512	02	04	46	87
2048	04	02	C8	FF		1024	03	02	C8	FF
4096	05	01	C8	FF		2048	04	01	C8	FF
MFM-Modus										
256	01	1A	0E	36	IBM-Disk.2D	256	01	12	0A	0C
512	02	0F	1B	54		256	01	10	20	32
1024	03	08	35	74	IBM-Disk.2D	512	02	08	2A	50
2048	04	04	99	FF		1024	03	04	80	F0
4096	05	02	C8	FF		2048	04	02	C8	FF
8192	06	01	C8	FF		4096	05	01	C8	FF

[1] Empfohlene Werte für GPL in Lese- oder Schreibbefehlen zur Vermeidung einer Überlappung von Datenfeld und ID-Feld benachbarter Abschnitte
[2] Empfohlene Werte für GPL in Formatierungsbefehlen

Bild 4 Marken des U 8272 D für FM (a) und MFM (b)

128 Byte/Sektor MFM vom U 8272 D zu realisieren. Es sind einseitig und doppelseitig beschreibbare Disketten durch den FDC-Schaltkreis ansteuerbar.

Aufzeichnungsdichte

Bei einfacher Aufzeichnungsdichte (Frequenzmodulation, FM) wird nach einem Taktimpuls entweder ein Datenimpuls (1) oder kein Datenimpuls (0) aufgezeichnet. Damit liegt bei einer 1 die doppelte Impulsfrequenz wie bei einer 0 vor. Die Taktimpulse dienen der Synchronisation beim Lesen der Daten, insbesondere, wenn viele Nullen nacheinander auftreten. Physisch werden die Takt- und Datenbits auf der Diskette als Wechsel der Flußrichtung realisiert. Enthält die Bitzelle (bestehend aus Takt- und Datenbit) eine 0, so ändert sich die Magnetisierung einmal, bei einer 1 zweimal pro Bitzelle. Die minimale Ausdehnung einer Bitzelle auf der Diskette ist durch die sichere Speicherung und Lesbarkeit begrenzt. Eine Erhöhung der Datendichte ohne Erhöhung der physischen Aufzeichnungsdichte erreicht man, indem man die Taktimpulse einfach wegläßt und nur bei mehreren Nullen ab der zweiten Null den Taktimpuls sendet (modifizierte Frequenzmodulation, MFM). Durch eine Verdopplung der Aufzeichnungsgeschwindigkeit (doppelter Schreibtakt des U 8272 D) erreicht man bei gleicher physischer Aufzeichnungsdichte die doppelte Datendichte. Der FDC-Schaltkreis realisiert beide Verfahren.

Aufzeichnungsformat

Das Aufzeichnungsformat des U 8272 D unterscheidet sich für Normal- und Minidisketten nur durch die Frequenz von C (Takt) und WRC (Schreibtakt). Bei Normaldisketten wird mit C = 8 MHz und WRC = 1 MHz (MFM) bzw. WRC = 0,5 MHz (FM) gearbeitet, während bei Minifloppylaufwerken C = 4 MHz und WRC = 0,5 MHz (MFM) bzw. 0,25 MHz (FM) gilt.
Die Formate für FM und MFM sind in Bild 3 angegeben. Zu beachten ist, daß der U 8272 D beim Formatisieren einer Spur auf die Vorderflanke des Indeximpulses (unabhängig von dessen Länge) reagiert und nach einer bestimmten Reaktionszeit mit der Ausgabe der Gap 4a (letzter Teil der Vorindexlücke) reagiert. Die Synchronfelder dienen zum Einsynchronisieren des U 8272 auf das Takt- und Datenraster, und die Marken dienen zur Kennzeichnung der nachfolgenden Felder.
Das Identifikationsfeld enthält die Information über Spur, Sektornummer und Byteanzahl des Sektors. Diese Informationen werden beim Formatieren einer Spur für jeden Sektor vom Prozessor abgefragt und sind demzufolge beliebig. Dadurch ergibt sich prinzipiell die Möglichkeit, die Sektoren in einer anderen Reihenfolge als mit steigender Nummer anzuordnen. Die CRC-Bytes werden vom U 8272 D entsprechend dem Polynom
$x^{16} + x^{12} + x^5 + 1$
gebildet und bei jedem Lesebefehl geprüft (auch bei einem Abbruch mitten im Datenfeld durch TC = 1 bzw. bei DTL < 128 Bytes für den laufenden Sektor). Die Gap 3 am Ende eines jeden Sektors ist durch Angabe des Wertes GPL (Gapelänge) im Befehl FORMAT A TRACK variabel. Zu beachten ist aber, daß bei den Lese- und Schreibbefehlen ein geringerer Wert für GPL angegeben wird, weil es sonst wegen der Toleranzen von Umdrehungsgeschwindigkeit und Taktfrequenzen zu Komplikationen kommen kann.
Tafel 1 zeigt für alle realisierten Formate des U 8272 D die empfohlenen Werte für GPL. In dieser Tafel sind außerdem die Beziehungen zwischen den Sektorgrößen und der Anzahl der möglichen Sektoren pro Spur angegeben. In Bild 3 sind für alle Felder nur die Werte der Datenbits angegeben. Für die Taktbits gilt bei FM der Wert FF und für MFM derjenige entsprechend dem Bildungsgesetz (siehe Abschnitt Aufzeichnungsdichte). Bei Marken gilt diese Regelung jedoch nicht, um eine Verwechslung mit Daten zu vermeiden. Die genaue Realisierung der Marken ist in Bild 4 gezeigt. Dabei wird das Taktbit mit dem dazugehörigen Datenbit zu einer Bitzelle zusammengefaßt, die für Normaldisketten bei FM eine Dauer von 4 μs (WRC = 0,5 MHz), bei MFM eine Dauer von 2 μs (WRC = 1 MHz) hat. Das Taktbit wird vor dem Datenbit gesendet und ist in Bild 4 schraffiert dargestellt. Die Hexadezimalwerte der ineinander verschachtelten Takt- und Datenbits sind nach den Impulsen angegeben. Bei FM wird nur 1 Byte für die Marken verwendet, während bei MFM jede Marke aus 4 Bytes besteht.

Kompatibilität

Der U 8272 ist bezüglich des Aufzeichnungsformates und der Funktion kompatibel mit den Schaltkreisen 8272A (INTEL) und μPD 765 A (NEC). Es können mit dem U 8272 auch problemlos Diskettenformate gelesen werden, die keine Indexmarke und keine Nachindexlücke besitzen (bei dem μPD 765 und INTEL 8272 war das nicht ohne weiteres möglich).

Tafel 2 Pinbelegung des U 8272

Pin-Nr.	Symbol	Typ	Name und Funktion
1	$\overline{R}$	I	RESET: Erzeugung des Grundzustandes
2	$\overline{RD}$	I	READ: Steuersignal für Datenübertragung
3	$\overline{WR}$	I	WRITE: Steuersignal für Datenübertragung
4	$\overline{CS}$	I	CHIP SELECT: Schaltkreisanwahl
5	AO	I	AO: Daten- u. Statusregisteranwahlsignal
6–13	DBØ-DB7	I/O	Datenbus
14	DRQ	O	DMA-REQUEST: DMA-Anforderung[1]
15	DAK	I	DMA-ACKNOWLEDGE: DMA-Bestätigung
16	C	I	TERMINAL COUNT: Beendigung des DMA-Transfers
17	IDX	I	INDEX: Anzeigen des Beginns einer Spur
18	INT	O	INTERRUPT: Interruptanforderung des FDC
19	C	I	CLOCK: Einphasen-8 MHz-Rechteck-Takt
20	USS		Masse
21	WRC	I	WRITE CLOCK: Datenschreibtakt
22	DW	I	DATA WINDOW: Datenbegleitsignal vom FDD
23	RDD	I	READ DATA: Daten vom FDD
24	VCO	O	VCØ: Regelung des Oszillators der PLL
25	WE	O	WRITE ENABLE: Freigabesignal zum Datenschreiben
26	MFM	O	MFM: Datenschreibmodus
27	HDSR	O	HEAD SELECT: Kopfauswahl
28, 29	DS1,DSØ	O	DRIVE SELECT: Auswahlsignal für FDD-Station
30	WRD	O	WRITE DATA: Schreibleitung der seriellen Takt- und Datenbits zum FDD
31, 32	PS1,PSØ	O	PRECOMPENSATION: Schreiben des Vorabgleichstatus bei MFM
33	F/TØ	I	FAULT/TRACKØ: Fehlermeldung des FDD/Spur 0
34	WP,TS	I	WRITE PROTECT/TWOSIDE: Prüfung auf Schreibschutz/zweiseitige Diskette
35	RDY	I	READY: Bereitschaft des FDD
36	HDL	O	HEAD LOAD: Laden des Lese-Schreibkopfes
37	FR/ST	O	FAULT RESET/STEP: Rücksetzen d. Fehler-FF in FDD/Schrittimpulse für die Kopfposition
38	LC/DR	O	LOW CURRENT/DIRECTION: Reduzieren des Kopfschreibstromes/Festlegung Kopfbewegung
39	$\overline{RW}$/S	O	READ-WRITE/SEEK: Auswahl des Betriebsmodus
40	UCC		Betriebsspannung +5 V

Für den U 8272 D ist zusätzlich zu den Formaten des μPD 765 A und INTEL 8272A das Format 128 Byte/Sektor MFM in Vorbereitung. Bei diesem Format wird keine variable Datenlänge (DTL) möglich sein.

Vorteile beim Einsatz des U 8272

Gegenüber herkömmlichen Lösungen bietet der U 8272 zahlreiche Vorteile:
1. Der Hardware-Aufwand sinkt auf ca. 60%.
2. Der Software-Aufwand (physische Basis-MOS) sinkt auf ca. 20%.
3. Der Aufwand für die Formatierung von Disketten reduziert sich auf ca. 60%.
4. Reduzierung der Zugriffszeit für jeden bearbeiteten Sektor um ca. 7 ms durch interne CRC-Berechnung.
5. Verringerung der BUS-Belastung (z. B. bei ID-Feld-Suche).

Mit diesen Vorteilen ist eine erhebliche Energie-, Material- und Volumeneinsparung verbunden.

Einsatzbedingungen und Herstellungstechnologien

Der U 8272 ist geeignet für den Einsatz in Verbindung mit zahlreichen CPU-Typen, einschließlich U 880, 8080A und 8086. Das Chip wird in einer nSGT-Technologie hergestellt und in einem 40poligen DIL-Plastgehäuse mit metrischem Pin- und Reihenabstand montiert. Die Betriebsspannung beträgt +5 V.
Ein Schaltkreis kann maximal 4 Floppy-Disk-Laufwerke bedienen, auf denen parallele Suchoperationen laufen können. Der Datenaustausch ist im DMA- und Nicht-DMA-Modus möglich, und es sind Mehrfachspur- und Mehrfachsektorübertragungen durchführbar.
Durch den U 8272 werden Steuersignale bereitgestellt bzw. verarbeitet, die den Aufwand für die externe PLL-Schaltung und die Vorkompensation vereinfachen. In den meisten Parametern stimmt der U 8272 mit dem i 8272A und μPD 765 A überein; er ist mit diesen Typen kompatibel in Pinbelegung und Funktion.
Die Pinbelegung mit einer kurzen Beschreibung ist in Tafel 2 zusammengefaßt.
Einen groben Überblick über die prinzipielle Funktion des U 8272 vermittelt das Blockschaltbild (Bild 5).
Der U 8272 enthält zwei über den Datenbus anwählbare Register. Bei $A_0 = 1$ liegt ein Datenregister an, das von der CPU gelesen und beschrieben werden kann. Das Hauptstatusregister bei $A_0 = 0$ kann von der CPU nur gelesen werden. Die Bits 0 bis 3 dieses Statusregisters kennzeichnen, welche der 4 angeschlossenen Laufwerke sich in einem Suchmodus befinden. Das Bit 4 wird gesetzt, wenn der FDC einen Lese- oder Schreibbefehl ausführt; dabei ist das Bit 5 gesetzt, wenn der Befehl im Nicht-DMA-Modus ausgeführt wird. Die Bits 6 (Direction) und 7 (Request for Master) kennzeichnen, ob der FDC zu einem Datenaustausch bereit ist und in welcher Richtung der Datenaustausch erfolgen soll.
Das Statusregister ist vor jeder Lese- und Schreiboperation der CPU auszuwerten (außer in der Ausführungsphase eines Befehls). Für die Reaktion auf das RQM-Flag stehen

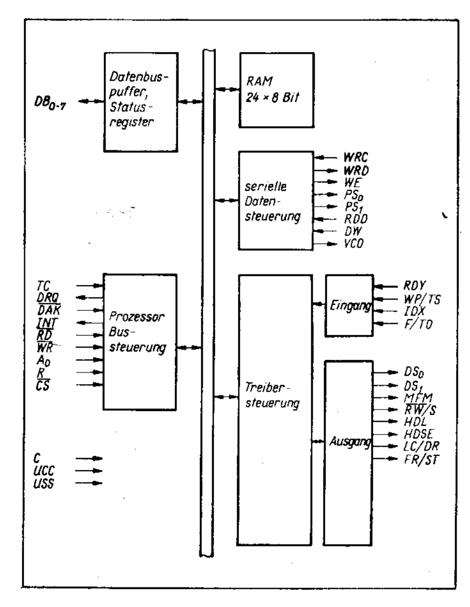


Bild 5 Blockschaltbild des U 8272

maximal 12 μs (bei 8 MHz) bzw. 24 μs (bei 4 MHz) zur Verfügung. Zu beachten ist weiterhin, daß dafür gesorgt werden muß, daß keine verbotene Belegung an die Pins A0, $\overline{RD}$ und $\overline{WR}$ gelegt wird, wie $\overline{RD} = 0$ und $\overline{WR} = 0$ bzw. $A_0 = 0$ und $\overline{WR} = 0$.

wird fortgesetzt

[1] Das DRQ-Pin des Schaltkreises U 8272 wird für verschiedene Testzwecke als Eingangspin benutzt. Um eine ordnungsgemäße Funktion des Schaltkreises unter allen Bedingungen zu gewährleisten, ist zwischen diesem Pin und U_{cc} ein Widerstand von 4,7 kΩ anzubringen.

Indizierte Variablen unter REDABAS

Das Datenbankprogramm REDABAS gestattet die Verwendung von bis zu 64 einfachen Variablen. Im Konzept von REDABAS sind allerdings keine indizierten Variablen oder Felder vorgesehen. Mitunter ist es aber notwendig, derartige Variablentypen zu benutzen. Bei entsprechender Programmierung ist es möglich, auch bei REDABAS Variablen zu schaffen, die wie indizierte verwendet werden können.
Der REDABAS-Interpreter verfügt über die Möglichkeit der Verwendung von Makros. Weist man einer Variablen (INDEX) ein Zeichen oder eine Zeichenkette zu und verknüpft den Namen der zu indizierenden Variablen (A) mit der Variablen INDEX mit Hilfe des Makrooperators in der Form A&INDEX, so wird durch den Interpreter bei der Abarbeitung anstelle von &INDEX der Inhalt der Variablen INDEX eingetragen. Auf diese Weise ist eine Indizierung von Variablen möglich.
Der Inhalt von INDEX muß in Verbindung mit dem vorangestellten Variablennamen stets den Konventionen für Speichernamen genügen. Aus diesem Grund ist im ersten Programmbeispiel auch eine Unterscheidung in Indexwerte kleiner 10 und größere nötig (Bild 1, Programm INDEX).
Im ersten Programmbeispiel werden 10 Variablen (A1–A10) mit frei wählbaren Inhalten belegt und im zweiten Teil des Programms nach Eingabe des Index angezeigt.
Im zweiten Programmbeispiel wird mittels der gleichen Vorgehensweise eine Matrix aufgebaut (Bild 2, Programm MATRIX).
Wird beim Aufruf von derartig indizierten Variablen ein unzulässiger Index verwendet, wird vom Interpreter ein Syntaxfehler gemeldet.
Beim Verfasser wird mit Hilfe der beschriebenen Indizierung innerhalb eines größeren REDABAS-Programmes ein einfacher „Tischrechner“ simuliert, der auch die Lösung einfacher linearer Gleichungssysteme ermöglicht.
Dadurch sind neben der der Datenband-Verwaltung auch einfache Kalkulationen möglich.

Bernd Matzke

```
* Programm INDEX
* Dieses Programm demonstriert die Verwendung von
* indizierten Variablen unter REDABAS
*
set talk off
erase
store 1 to ZAEHLER
do while ZAEHLER<=10
    if ZAEHLER<10
        store str(ZAEHLER,1) to INDEX
    else
        store str(ZAEHLER,2) to INDEX
    endif
    accept  Inhalt der Variablen eingeben' to A&INDEX
    store ZAEHLER+1 to ZAEHLER
enddo
display memory
do while T
    accept 'Index eingeben' to INDEX
    ? A&INDEX
enddo
return
```

```
* Programm MATRIX
* Dieses Programm demonstriert die Arbeit mit
* einer Matrix unter REDABAS
*
erase
set talk off
store 1 to B
do while B<=5
    store str(B,1) to Y
    store 1 to A
    do while A<=7
        store str(A,1) to X
        store chr(59+B+5*A) to MEM&X&Y
        store A+1 to A
    enddo
    store B+1 to B
enddo
display memory
do while T
    accept 'Eingabe Zeile' to X
    accept 'Eingabe Spalte' to Y
    ? MEM&X&Y
enddo
return
```

MC 80/20 mit K-5221-Magnetbandlaufwerk und 48-K-RAM

Wird der MC 80 als variables Entwicklungssystem genutzt, treten bei der Originalspeicherkonfiguration einige Probleme auf. Eine Speicherorganisation gemäß Tafel 1 hat sich dann als günstig erwiesen.
Kernstück dieser Speicherversion ist eine mit geringem Aufwand umgerüstete 16-K-EPROM-Steckeinheit K 3820. Es wurde eine Möglichkeit gefunden, zwei schaltbare, adreßparallele 8-K-Blöcke zu bilden. Damit lassen sich mit dieser weit verbreiteten Steckeinheit „ROM-Floppy-Systeme" aufbauen. Im beschriebenen Anwendungsfall sind im Vordergrund der Editor des MC 80, im Hintergrund der 5 K lange Magnetbandtreiber angeordnet. Da bei der Arbeit mit diesem Treiber (Hersteller VEB Elektronik Gera) aus Zeitgründen eine Display-Abschaltung erfolgt, kann ohne Nachteile während dieser Zeit der parallel liegende EDITOR stillgelegt werden.
Die Vorteile dieser Variante sind, neben der bequemen Arbeit mit dem Magnetbandlaufwerk, eine optimal baugruppenbezogene Speicherausnutzung und das Laden von Programmsystemen nach Bedarf, wie BASIC, TEXTEDITOR, EDITOR/SIMULATOR U 882 usw., ab 4000H. Mit geringfügigen Änderungen im Betriebssystem wird es möglich, diese Programme auch in der Menütabelle auszugeben und so zu starten.

Tafel 1

0000H	ZRE	4 K Originalbelegung (BSYS)	
1000H		4 K Freibereich[1]	
2000H	PFS K3820 in 2×8K Organisation	Vordergrund	Hintergrund
		1 K Freibereich[1]	5 K K-5200-Treiber
2400H		1 K COPY-Treiber	3 K Freibereich
2800H		6 K Originalbelegung (EDITOR)	
4000H FFFFH		48 K durchgängig RAM-Anwenderspeicher	

[1] Diese Freibereiche können z. B. zum Nachrüsten einer Vollgrafik auf der Basis des modularen Systems QFD 1520/GSP 1520 /1/ genutzt werden.

Folgende Hard- und Softwareänderungen sind vorzunehmen:

1. Initialisierung der ZRE-PIO, Port B0 auf Ausgabe für MEMDI1-Generierung der umgerüsteten PFS K 3820; das Programmstück wird im unbenutzten Betriebssystembereich angelegt (Tafel 2).
2. Zum Eintrag „K 5200" in das Originalmenü auf 3FFFH steht kein Platz zur Verfügung, deshalb wird diese Tabelle auf 27FFH (COPY-EPROM) verlegt, die alte Tabelle auf 3FE8H-3FFFH kann gelöscht werden (Tafel 3).
3. Im Original-Betriebssystem wird der Speicherplatz 7FFFH auf eine Programmtabelle

Tafel 2

BETRIEBS-SYSTEMBE-REICH S.01		(000H-03FFH)
0069H	JMP 0B04H	ANSPRUNG ERWEITERUNG
006DH	....	ORIGINAL
BETRIEBS-SYSTEMBE-REICH S.03		(0800H-0BFFH)
0B04H	LD A, 0FFH	PIO-INITIALISIERUNG
0B06H	OUT 87H	ZRE-PIO B MODE 3
0B08H	LD A, 0FEH	
0B0AH	OUT 87H	B0: AUSGANG B1-7: EINGAENGE
0B0CH	LD A,01H	B0: 1; 1. 8 K-BLOCK (EDITOR) AKTIV
0B0DH	OUT 85H	
0B10H	LD A, 0DH	ORIGINALBEFEHLE VON 0069H
0B12H	LD I,A	
0B14H	JMP 006DH	NORMAL WEITER
0B17H	XOR A	MENUE-ANSPRUNG K 5200
0B18H	OUT 85H	EDITOR AUS, 2. 8K-BLOCK AUF 2000 MIT K 5200-TREIBER AKTIV
0B1AH	JMP 2ED9H	SPRUNG AUF AKTIVIERTEN TREIBER

Der Treiber wird mit RST 0 verlassen, damit Speicherblockumschaltung (Editor ein).

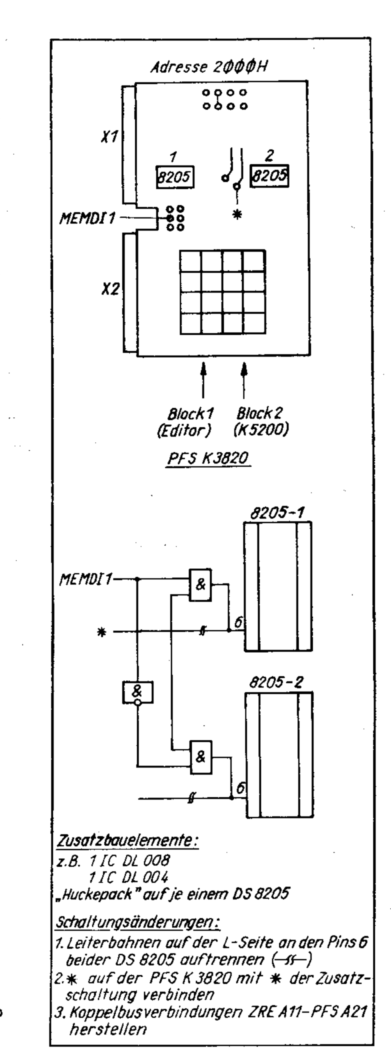


Bild 1 Zusatzschaltung und Schaltungsänderung auf der PFS K 3820 zur Speicherblockumschaltung ▶

Tafel 3

BETRIEBSSYSTEM-BEREICH S.02		*(0400H-07FFH)*
04C1H	*LD HL,27FFH*	*NEUE PROGRAMMTABELLE 3*
INHALT NEUE TABELLE:		
	DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE	
27DF:	*C7 4F 52 50 45 80 00 28 CD 41 52 8 0 BE 2E D2 4F*	
27EF:	*54 49 44 45 80 00 30 B0 30 32 35 20 4B 80 17 0B*	
27FF:	*00*	

getestet. Für die vorliegende Speicherkonfiguration ist das nicht sinnvoll, deshalb wird die Tabelle beispielsweise auf 400FH verlegt. Damit wird erreicht, daß kurze Programme, die nicht auf 5FFFH eingetragen werden können, unter Nutzung dieser Tabelle angezeigt werden. Bei nichtbeabsichtigter Nutzung der Prorammtabellen auf 400FH bzw. 5FFFH hat der Programmierer dafür zu sorgen, daß auf diesen Plätzen nicht „00" steht.

Betriebssystembereich S.02 (0400H-07FFH)
04E1H LD HL,400FH Neue Programmtabelle 4

4. Bei vorliegender Speicherkonfiguration sollte die automatische Abfrage der Adresse 6000H (Vorhandensein eines RESET-Initialisierungsprogrammes) unterbunden bzw. dafür eine andere Adresse gewählt werden, z. B.:
00B3H JMP 04ACH Überspringen der Abfrage

5. Die Schaltungsänderung der PFS K 3820 hat entsprechend Bild 1 zu erfolgen. Die Realisierung der Torschaltung sowie die Einspeisung der Blockselektierung ist beispielhaft angegeben. Die Leiterzüge an den Pins 6 der beiden DS 8205 sind in jedem Fall aufzutrennen. Der auf 2000H gebundene und verbesserte K-5200-Treiber sowie das geänderte Betriebssystem können nachgenutzt werden.

Dr. Hans-Ulrich Stiehl

Literatur

/1/ Hartmann, M.; Hiller, H.; Mensel, K.-H.: Modulares Farbgrafiksystem GFD 1520/m und seine Anwendungen. Radio, Ferns., Elektron., Berlin 36 (1987) 2, S. 71–77

✉ KONTAKT ☎

Technische Universität Dresden, Sektion Chemie, WB Technische Chemie, Mommsenstr. 13, Dresden, 8027; Tel. 4 63 38 19

Sichtsystem mit CCD-Zeilenkameras

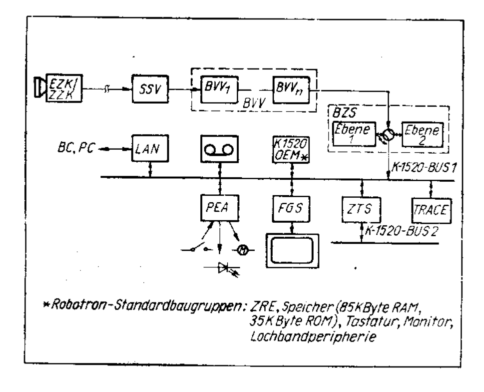
Dr. Axel Röhl, VEB Fischfang Rostock
Dr. Klaus-Peter Schulz, Prof. Dr. Otto Fiedler, Dr. Heinrich Albrecht, Wilhelm-Pieck-Universität Rostock

1. Einführung

Bei der Überwachung kontinuierlich ablaufender Produktionsprozesse müssen häufig bewegte Objekte vermessen und sortiert werden. Das Abtasten der Objekte in einer Ebene kann günstig mit Hilfe von CCD-Sensorzeilen vorgenommen werden, insbesondere, wenn die Bewegung gleichförmig in einer Richtung erfolgt (Bandförderung). Ist die Bewegung ungleichförmig, z. B. bei Schlupf des Objektes gegenüber dem Transportband, so wird eine Geschwindigkeitssensierung erforderlich. Das ist mit einer zweiten CCD-Zeile möglich /1/. Das CCD-Meßkonzept garantiert eine stabile und verschleißfreie Objektabtastung bei hoher örtlicher und zeitlicher Auflösung und eine gute Anpassung an unterschiedliche Meßaufgaben. Der technische Aufwand wächst mit den Anforderungen an Genauigkeit und Flexibilität des Systems.

Eine Grauwertverarbeitung bietet gegenüber der einfachen Binärbildverarbeitung die Möglichkeit, bestimmte Merkmale mit Hilfe von Vorverarbeitungsalgorithmen hervorzuheben. Häufig sind es Kanten oder Grenzlinien, auf die Bezug genommen wird. Um sie zu verstärken und hervorzuheben, wendet man z. B. Kombinationen von Glättungs- und Versteilerungsoperationen an.

Bild 1 Blockschaltbild des Sichtsystems



2. Technische Realisierung

2.1. Gesamtsystem

Zur On-line-Erfassung von Objekten bezüglich Muster und Geschwindigkeit wurden CCD-Kameras mit Ein- und Zweizeilen-Anordnungen entwickelt und eingesetzt. Ihre Signalausgänge sind einheitlich und wahlweise an die nachfolgenden Auswertebaugruppen anschließbar. Eine Kamera wird gemäß Bild 1 über die Baugruppen

– Signalvorverarbeitung SVV
– Bildvorverarbeitung BVV (bei Bedarf) und
– Bildzwischenspeicher BZS

mit dem Mikrorechner K1520 (Systembus) verbunden.

Ergänzende Baugruppen neben dem K1520-Standardsortiment sind:

– Zweitorspeicher ZTS und
– Farbgrafiksteuerung FGS zum Einfahren des Systems
– potentialtrennende Ein-/Ausgabe PEA für den Anschluß von Initiatoren, Stellmotoren, Signallampen und -Tastern sowie
– Anschlußbaugruppe zum LAN und
– TRACE-Baugruppe zur Unterstützung der Programmierung.

Bild 1 kennzeichnet das an der Sektion TE der WPU entwickelte und installierte Sichtsystem, das ohne LAN, ZTS und Lochbandanschluß als Meß- und Steuersystem in Maschinen zur automatisierten Fischverarbeitung mit Erfolg getestet wurde.

2.2. Beleuchtung

Die Beleuchtung und Szenengestaltung im Beobachtungsraum sind wesentliche Faktoren im Prozeß der Merkmalerkennung. Die Realisierung erfolgt aufgabenspezifisch nach folgenden Gesichtspunkten:

– Beleuchtungsverfahren (Durchlicht, diffuses oder gerichtetes Auflicht, Lichtschnitt, Fluoreszenz)
– Niveau und örtliche Verteilung der Beleuchtungsstärke
– zeitliche Änderungen der Beleuchtungsstärke
– spektrale Leuchtdichteverteilung (Filtereinsatz) und
– Stabilität und Schutz der Beleuchtungseinrichtung (Spritzwasser, Korrosion, Lebensdauer und Glühlampen).

Für den Einsatz in der Fischverarbeitung kamen Durchlicht- und Lichtschnittverfahren nicht in Frage. Bewährt haben sich:

a) Anordnungen einzeln justierbarer Halogenleuchten mit Hohlspiegel und Mattglasscheibe über diffus reflektierendem Untergrund (umgebaute Kfz-Nebelscheinwerfer mit Halogen-Glühlampe des Typs H3 zu 24 V/70 W) und

b) Lampenkästen mit 4 oder 6 aneinandergereihten Reflektorlampen vom Typ HLLR (12 V/20 W) zur Erzeugung eines gerichteten Auflichts.

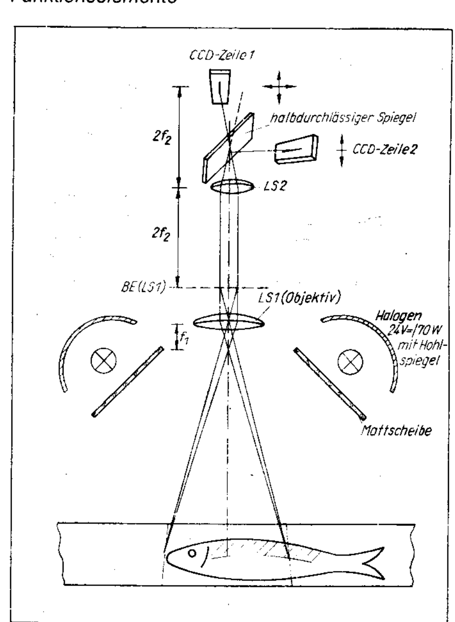
Mit einer Versuchsanordnung nach a) wurden gleichmäßige, flächenhafte Ausleuchtungen für den Einsatz einer Zweizeilenkamera (s. Bild 2) erzielt. Die Variante b) wurde für den Betrieb mit einer Einzeilenkamera im Reflexionswinkel über ebenem, metallischglänzendem Untergrund entwickelt.

Durch Speisung über einen Sechspuls-Brückengleichrichter wurden hinreichend kleine Flimmerfaktoren von 0,78 % zu a) und 0,36 % zu b) erzielt.

2.3. CCD-Zeilenkameras

Der für die meisten Anwendungsfälle entscheidende Leistungsparameter des Sichtsystems ist die Ortsauflösung. Sie wird nicht nur vom optoelektronischen Wandler, son-

Bild 2 Schematische Anordnung der optischen Funktionselemente



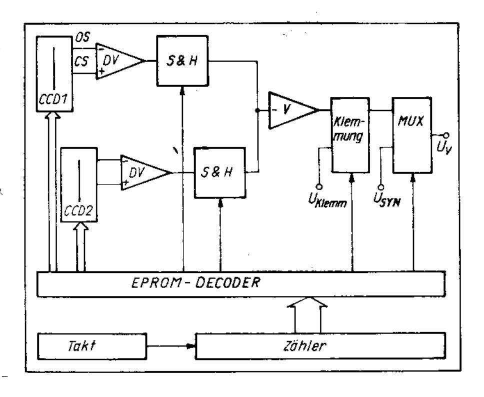


Bild 3 Blockschaltbild der Zweizeilenkamera ZZK2

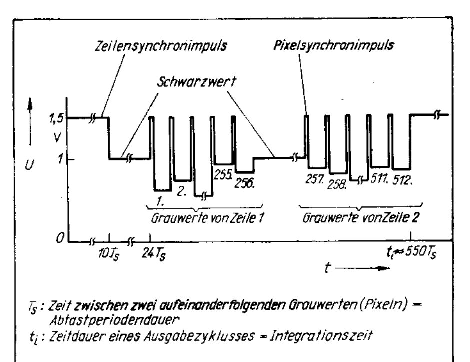


Bild 4 Video-Ausgangssignal der Zweizeilenkamera

dern auch von den vor- und nachgeordneten Baugruppen wie Beleuchtung, Objektiv und Videoverstärker bestimmt. Eine zusammenfassende Betrachtung ist durch die Anwendung der Systemtheorie in der abbildenden Optik möglich. Analog zum Amplitudenfrequenzgang von Zeitfunktionen wird die Modulationsübertragungsfunktion MTF (Modulation Transfer Function) von Ortsfunktionen ermittelt. Bei der MTF-Analyse wird unterschieden zwischen Anteilen, die vor bzw. nach dem eigentlichen Abtastvorgang wirksam werden. Erstere ermöglichen in ihrer Gesamtheit eine Aussage zur Einhaltung des Abtasttheorems. Da Unterabtastung zu Fehlinterpretationen durch Scheinkonstruktionen (aliasing) führen kann, muß mittels einer geeigneten Parameterauswahl (z. B. Objektgeschwindigkeit, Integrationszeit, Infrarotanteil der Beleuchtung) die Bedingung des Abtasttheorems eingehalten werden. Umfangreiche theoretische und praktische Untersuchungen zur MTF sind in /1/ durchgeführt worden.
Definiert man das maximale Auflösungsvermögen als die Ortsfrequenz, bei der die MTF um 3 dB abgefallen ist, so ergeben sich für eine Einzeilenkamera EZK3 ca. 10 Linienpaare/mm (Objektiv Tevidon 2/10, Blendenzahl k = 2,8...4) und für eine Zweizeilenkamera ZZK2 ca. 7 Linienpaare/mm (Tevidon 2/10, k = 4...5,6) in Zeilenrichtung. Die Messungen erfolgten bei Beleuchtung mittels Halogenlampen hinter einer Mattglasscheibe und ohne Einsatz eines Infrarot-Sperrfilters.
Die eingesetzten Ein- und Zweizeilenkameras sind in Kompaktbauweise mit CCD-Zeilen L110C bzw. L133C (VEB Werk für Fernsehelektronik Berlin) ausgeführt. Entsprechend den Einsatzbedingungen wurde eine konstruktive Lösung entwickelt, bei der Kameraelektronik und Objektiv (Tevidone oder Practica, VEB Carl Zeiss JENA) vollständig gekapselt sind.
Das einheitliche Gesamtkonzept soll am Beispiel der Zweizeilenkamera ZZK2 nach /2/ erläutert werden. Die Abbildung erfolgt über ein auswechselbares Objektiv, ein zweites Linsensystem (LS2) und einen Strahlteiler auf die beiden CCD-Sensor-Zeilen (L110C), wie Bild 3 anhand eines Einsatzfalles skizziert.
Die Ausgangssignale der CCD-Zeilen an OS (Output Source) und die Kompensationssignale an CS (Compensation Source) zur Unterdrückung des Rückstelltaktrauschens gelangen über Emitterfolger an einen integrierten Differenzverstärker (DV), der eine Anhebung des Grauwertsignales um ca. 28 dB bei hoher Gleichtaktunterdrückung von ca 100 dB bewirkt. Den Abtast- und Halteschaltungen (S&H), die gleichzeitig als Multiplexer fungieren, folgt ein Kaskodeverstärker mit einstellbarer Verstärkung von 0...15 dB. Nach Schwarzwertklemmung und Einmischen der Synchronsignale ergibt sich am unsymmetrischen Ausgang mit 75 Ohm Abschlußimpedanz das in Bild 4 skizzierte Videosignal.
Die erreichbare Datenrate fs der ZZK2 wird durch die Zugriffszeit des Decoder-EPROMs auf 1,25 MHz begrenzt.

2.4. Signalvorverarbeitung SVV

Durch die Verwendung eines BAS-ähnlichen Kompaktsignales werden Laufzeitprobleme an längeren Signalleitungen vermieden (bis 30 m erprobt). In der SVV wird das Signal verstärkt (Kompensation der Leitungsverluste), von den Synchronimpulsen getrennt und digitalisiert. Für den Industrieeinsatz steht eine 4-Bit-Parallel-Wandler-Baugruppe SVV4.3 mit Einzelkomparatoren und für Laborzwecke eine 6-Bit-Variante SVV6.1 mit ADU-Schaltkreis K1107PW1A zur Verfügung. Über eine Potentialtrennung (Optokoppler MB111) liegen das digitale Bildsignal und die Synchronsignale am Koppelbus an. Die Durchsatzraten von 6 MBit/s bei 4-Bit- und 10 MBit/s bei 6-Bit-Umsetzung sind ausreichend und stellen im Signalweg keinen Engpaß dar.
Für die Verwendung industrieller Zeilenkameras vom Typ ZFK 1021 (VEB Studiotechnik Berlin) wurde in Ergänzung zu der SVV4.3 die SVV4.4 geschaffen. Das Taktsystem wird durch einstellbare Teiler (Arbeitstakt: 2,5 MHz; 1,25 MHz; 625 kHz; Integrationszeit: 216 µs, 432 µs, 864 µs) aus dem K-1520-Systemtakt abgeleitet. Elektrische und mechanische Kompatibilität sind gewährleistet.

Tafel 1 Realisierbare Filteroperationen mit der Baugruppe zur Bildvorverarbeitung

Operator	Anzahl der Prozessoren	Kopplungsart
Rangordnungsoperator	1	–
Zweipunkt-Gradient	1	–
Kantenversteilerung	2	parallel
Verdünnung	2	seriell
Direktionaler Median	1	–
Randglättung	1	–

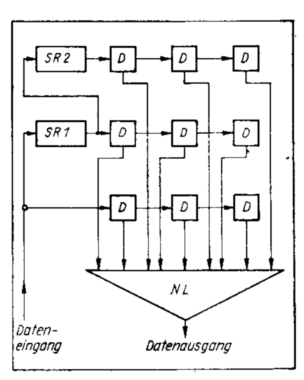


Bild 5 Nachbarschaftslogik in der BVV

2.5. Bildvorverarbeitung

Die Einplatinenbaugruppe BVV entstand durch Modifizierung des GIPP (Grayscale Image Preprocessor des Zentralinstitutes für Kybernetik und Informationsprozesse der AdW, s. a. /3/) zur Anpassung der Steuersignale und im Hinblick auf den Einsatz von Richttypenbauelementen. Dieser Nachbarschaftsprozessor bestimmt durch nichtlineare Verknüpfung der 3 × 3 Bildpunkte eines gleitenden Fensters die neue Belegung des aktuellen Zentralpunktes im Fenster. Mit der BVV sind die in Tafel 1 aufgeführten nichtlinearen Filteroperationen möglich. Bild 5 zeigt das vereinfachte Blockschaltbild.
Die Schieberegister SR1 und SR2 speichern 2 vollständige Zeilen von bis zu 1024 Elementen, so daß dem Prozessor mit der aktuellen Zeile insgesamt 3 benachbarte Zeilen zur Verfügung stehen. Jeweils 3 Grauwerte einer Zeile werden in den Verzögerungselementen D zwischengespeichert. Die Nachbarschaftslogik NL führt die parallele Verknüpfung aller 9 Grauwerte im Bildpunkttakt durch. Die 4 Bit des neuen Zentralpunktes werden durch sukzessive Annäherung zeitseriell bestimmt.
Für die Bildvorverarbeitung können mehrere Baugruppen in einer Pipeline oder auch parallel (je Zweig ein BZS notwendig) geschaltet werden (z. B. MEDIAN zur Rauschunterdrükkung und Zweipunkt-Gradient zur Kantenhervorhebung in Pipeline).
Abhängig von der Bestückung sind Pixelfrequenzen bis zu 1,25 MHz möglich. Die Synchronisation geschieht über die in der SVV zurückgewonnenen Zeilen- und Bildpunktsynchronsignale. Jede BVV bewirkt eine Verzögerung um eine Zeile und einen Bildpunkt.

2.6. Bildzwischenspeicher

Die Pipeline der Sichtsystembaugruppen wird in Richtung Verarbeitungsrechner durch einen zugeschnittenen Zweitorspeicher abgeschlossen. Er besitzt zwei voneinander unabhängige Speicherebenen zu je 1 K × 8 Bit. Eine Ebene wird im Bildpunkttakt mit der aktuellen Grauwertfolge beschrieben. Parallel dazu ist dem Rechner der freie, asynchrone Lesezugriff auf die andere Ebene möglich. Die Torumschaltung erfolgt mit dem ersten Lesezugriff auf die jeweilige Ebene. Der Speicher belegt 2 KByte im Adreßraum des Rechners (Quasi-ROM). Die Adresse für den Schreibzugriff entsteht durch Zählung der Bildpunktsynchronimpulse. Der Zeilensynchronimpuls setzt diese Zähler zurück.

2.7. Potentialtrennende Ein-/Ausgabe PEA

An diese auf eine spezielle Anwendung zugeschnittene Baugruppe können potentialgetrennt 6 induktive Näherungsinitiatoren, 3 Schrittantriebe NSA-S62 (mit Zusatzelektronik) und 6 BCD-Schalter angeschlossen werden. Weiterhin enthält diese Baugruppe eine NMI- und RESET-Logik und bietet Testmöglichkeiten für die BVV.

2.8. Farbgrafiksteuerung

Ein wichtiges Hilfsmittel für die subjektive

Beurteilung des digitalisierten Bildes ist eine farbige oder grauwertige Rasterdarstellung. Das gilt insbesondere für die heuristische Auswahl nichtlinearer, lokaler Operationen, da deren Wirkung auf das Ortsfrequenzspektrum objektabhängig ist. Die entwickelte Steuerung bietet eine Auflösung von 512 × 256 Bildpunkten zu 3 Bit (8 Falschfarben oder Graustufen) und belegt 3 × 16 KByte Speicherraum, die parallel zum Hauptspeicher liegen (Speicherbank, Umschaltung über MEMDI-Signale). Diese Lösung ist ein Kompromiß zwischen relativ geringem Aufwand und zusätzlicher Rechnerbelastung für die Grafikverwaltung.

2.9. Zweitorspeicher ZTS

Bei höherer Belastung der CPU im K 1520 mit aufwendigerer Meßsoftware muß z. B. die Überwachung und Steuerung der Stellmotoren einer zweiten Recheneinheit übertragen werden. Der ZTS bietet die Möglichkeit, auch umfangreichere Datensätze (z. B. geometrische Maße) zwischen zwei Rechnern auszutauschen. Analog zum BZS steht im ZTS für jeden Rechner eine Speicherebene (2 × 2 KByte, bidirektionaler Zugriff) zur Verfügung. Ein Rechner besitzt die Zugriffspriorität.

3. Einsatz des Systems

Der Ersteinsatz des beschriebenen Sichtsystems erfolgte in der Fischlängen-Feinsortierung an Bord eines Fang- und Verarbeitungsschiffes des VEB Fischfang Rostock. Die Sortierung gelang mit einem mittleren Fehleranteil von 3,3%, bezogen auf einen zulässigen Toleranzbereich der Klassengrenzen von 5 mm (absoluter Maximalfehler) bei einer Transportgeschwindigkeit von 1,35 m/s. Gegenwärtig wird der Einsatz des Sichtsystems in einer neuartigen Fischbearbeitungsmaschine erprobt. Die technologische Aufgabenstellung fordert bei einem Durchsatz von 260 Fischen/min einen optimalen Köpfschnitt unmittelbar am Kiemenbogen, um eine maximale Filetgröße zu erzielen. Der erste Lösungsansatz geht von einer indirekten Bestimmung der Kopflänge über eine lineare Gleichung der Art $K = a * G + b$ aus, wobei K die Kopflänge, G die gemessene geometrische Größe (Höhe, Länge o. ä.) und a, b artabhängige Parameter sind. Die Meßprogramme wurden in Assemblersprache geschrieben und zeitoptimiert. Hilfsprogramme gestatten Aussagen zur Szene (räumliche und statistische Grauwertverteilung) und zu den Stellmotoren (Stellgeschwindigkeit und Stellsicherheit). In einer ersten Variante wurde die maximale Bauch-Rücken-Höhe

(Fortsetzung auf S. 123)

Echtzeit-Softwareanalysegerät

Michael Jacoby, Ingenieurbetrieb für die Anwendung der Mikroelektronik Potsdam
Dr. André Rompe, Akademie der Wissenschaften der DDR, Institut für Kosmosforschung

Die gegenwärtige Entwicklung von Logikanalysatoren bestätigt den Trend zu immer ausgefeilteren Softwareanalysetechniken. Eine der kostspieligen Komponenten dieser Analysatoren sind die schnellen RAM-Speicher, die Abtastraten bis zu 2 GHz in modernen Geräten ermöglichen. Beschränkt man sich auf das Gebiet der Softwareanalyse, so resultiert eine drastische Reduzierung der erforderlichen RAM-Zugriffszeiten, da eine Abspeicherung auf Maschinenzyklusebene dazu ausreichend ist. Aufgrund der Tatsache, daß bei zahlreichen Entwicklungen der Softwareaufwand 50% und mehr des Gesamtaufwandes ausmacht, erscheint die Entwicklung spezieller Softwareanalysegeräte gerechtfertigt. Für die Entwicklung und Testung von U880-Software unter praxisnahen Bedingungen wurde vom VEB Mikroelektronik „Karl Liebknecht" Stahnsdorf (Ingenieurbetrieb für die Anwendung der Mikroelektronik Potsdam) und dem Institut für Kosmosforschung der Akademie der Wissenschaften ein autonomes Softwareanalysegerät für U880-Mikrorechnersysteme entwickelt.
Auf der Grundlage eines neuartigen Markierungsspeicherprinzips wurde eine selektive und prozeßorientierte Datenselektion und Abspeicherung möglich. Das auf diesem Prinzip entwickelte Gerät basiert ausschließlich auf in der DDR verfügbaren Bauelementen und kann sich in seiner Leistungsfähigkeit in bezug auf Aufgaben der Softwareanalyse international mit den leistungsfähigsten Logikanalysatoren messen.

Aufbau und Funktionsprinzip

Das Softwareanalysegerät besitzt einen eigenen U880-Mikrorechner für die Bedienung und Auswertung. Der Speicher besteht aus 16 KByte ROM und einer 8-KByte-SRAM-Einheit sowie der erforderlichen Analysatorsteuerlogik. Der Datenbus (DB) sowie der Adreßbus (AB) des zu analysierenden Systems und eine 3-Bit-Marke werden in einem 1 KByte tiefen Analysatorspeicher aufgefangen. Ein 4 KByte tiefer Markierungsspeicher, der mit Hilfe einer Adreßverschiebelogik über den gesamten Adreßbereich des U880 positioniert werden kann, gewährleistet die Auslösung und Abarbeitung der vorprogrammierten Analysekommandos während des Echtzeitlaufes. Die Ankopplung an das zu untersuchende System erfolgt über einen aktiven Buskoppler von ca. 1,5 m Länge. Es ist eine direkte Kopplung an die CPU mit einem Clip oder über den Systembus-Steckverbinder des zu testenden Systems möglich. Für das K1520-System existiert eine entsprechende Koppelkarte. Eine integrierte Tastatur und ein Monitor mit BAS-Anschluß ermöglichen die Kommunikation mit dem Analysator. Optional kann eine externe Robotron-Standardtastatur angeschlossen werden.
In Vorbereitung der Abspeicherung werden in ein Menübild die Programmadressen und je ein zugehöriges Analysekommando in mnemonischer Form eingetragen. Nach dem Abschluß aller Eintragungen werden in den Markierungsspeicher an den bezeichneten Adressen den Analysekommandos entsprechende Kontrollcodes vom Steuerrechner des Gerätes eingetragen. Der Markierungsspeicher wird mit Hilfe einer Logik adreßmäßig parallel zu dem zu testenden System während des Echtzeitlaufes geschaltet. So wird beim Erreichen der kommandierten Adressen der zugehörige Kontrollkode aus dem Markierungsspeicher ausgelesen und von einem Befehlsdekoder erkannt und ausgeführt. Bild 1 zeigt die prinzipielle Schaltungsanordnung. Der entscheidende Vorteil dieser Markierungstechnik gegenüber einer Adreßdekodertriggerung besteht darin, daß praktisch beliebig viele selektive Analysekommandos genutzt werden können. Auf dieser Basis wird eine gezielte und redundanzarme Datenselektion möglich.
Folgende Analysemöglichkeiten wurden mit Hilfe dieses patentrechtlich geschützten Verfahrens realisiert:

Debug-Mode

Selektive Abspeicherung von Daten- und Adreßinhalten beliebiger Befehle des zu untersuchenden Programms. Der oder die markierten Befehle werden maschinenzyklusweise im Analysespeicher aufgefangen. Eine Unterdrückung der Befehlsbytes bei der Abspeicherung ermöglicht die redundanzfreie Aufzeichnung ausschließlich der nicht zum Befehlskode gehörenden Datenzyklen. Die Identifikation der aufgezeichneten Daten erfolgt über eine Marke bzw. über die mitaufgezeichnete Programmadresse. In dieser Betriebsart ist eine Aufzeichnung und spätere Auswertung von den Prozeßzustand charakterisierenden Rechnergrößen möglich.
Mit Hilfe des 2-Kanal-DA-Wandlers können zwei voneinander unabhängige Rechengrößen, die Zustandsgrößen, Meßwerte, Zwischenergebnisse oder anderes sein können, in Echtzeit oder auch bei der Datenauswertung analog auf einem entsprechenden Gerät, z. B. einem Oszillographen oder Digitalvoltmeter, soweit es sich um eine nur langsam ändernde Größe handelt, dargestellt werden. Es ist bedeutsam, darauf hinzuweisen, daß in keinem Fall der Ablauf des zu untersuchenden Rechnersystems in irgendeiner Weise beeinflußt wird. Dies ist besonders für zeitkritische Echtzeitanwendungen wesentlich.
Für die Anwendung des Debug-Modes sei folgendes Beispiel gegeben: Eine Regelstrecke liefere die Meßwerte Xi, die AD-gewandelt vom Port A des Mikrorechners eingelesen werden. Durch Filterung der Meßwerte entstehen die Schätzwerte Xi. Diese werden mit einem Speicherreferenzbefehl im Arbeitsspeicher des Reglerprogramms abgelegt. Im weiteren Regelprogramm seien die Zwischengrößen Z1 und Z2 von Interesse. Den prinzipiellen Verlauf des Programms zeigt Tafel 1.
Im Vorbereitungsbild werden bei den interessierenden Adressen die gewünschten Analysekommandos eingetragen (DEB 1–DEB 5). Die Zahlen hinter dem Debug-Kommando sind Marken zur Kennzeichnung der aufgezeichneten Daten bei der Auswertung. Nach dem Echtzeitlauf des zu testenden Programms sind im Analysatorspeicher die Daten und zugehörigen Adressen der im Vorbereitungsmenü markierten Befehle einschließlich der zugehörigen Marken abgespeichert.
Im Vorbereitungsmenü kann eine Unterdrückung der Befehlszyklen der markierten Befehle und damit die ausschließliche Aufzeichnung der transportierten Daten eingestellt werden. Diese Möglichkeit ist für die analoge Darstellung der ausgezeichneten Daten er-

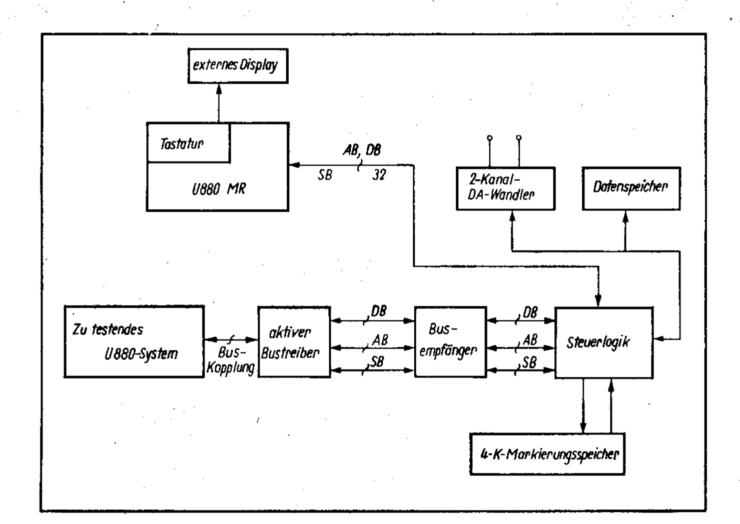


Bild 1 Autonomes Echtzeit-Softwareanalysegerät

Tafel 1 Beispiel zur Programmierung der selektiven Datenanalyse

Adresse	Mnemonik	Kommentar	Analyse Kommando
4444H	In A,(A)	Meßwert Xi	DEB 1
.			
4567H	LD (HL),A	In A steht Xi	DEB 2
.			
4678H	LD BC,(HL)	In (HL) steht Z1	DEB 3
.			
4789H	POP DE	Im Stack steht Z2	DEB 4
.			
489AH	OUT (B),A	A ist Stellgr.	DEB 5

forderlich und sichert eine redundanzfreie Datenaufzeichnung.
In einem seriellen Auswertebild des Analysators werden die aufgezeichneten Daten, Adressen und Marken hintereinander dargestellt. Interruptvektoren können in der Aufzeichnung wahlweise mitaufgezeichnet werden. Die Aufzeichnung kann so gesteuert werden, daß entweder die ersten 1023 oder die letzten 1023 Ereignisse aufgezeichnet werden (START und END TRACE). Weiterhin kann die Aufzeichnung erst nach dem n-ten Vorbeilauf an einer beliebigen Adresse gestartet werden (TRIGGER DELAY).

Selektive Abspeicherung

Eine weitere wichtige Analysemöglichkeit besteht in der selektiven Abspeicherung von beliebigen Programmsequenzen. Das bedeutet das *wiederholt* mögliche Ein- und Ausschalten der Aufzeichnung an beliebigen Programmstellen. So können z. B. interessierende Unterprogramme wie Interruptserviceroutinen bzw. bestimmte Programmausschnitte aufgezeichnet werden. Dadurch wird eine redundanzarme Programmabspeicherung und Analyse möglich, die eine bestmögliche Konzentration auf die zu untersuchenden Programmteile sicherstellt. Die Abspeicherung erfolgt dabei vom ersten markierten Befehl bis ausschließlich des letzten markierten Befehls. Dabei werden alle Maschinenzyklen aufgezeichnet, so daß bei der Auswertung eine disassemblierte Darstellung der aufgezeichneten Programmsequenzen genutzt werden kann. Mit Hilfe dieser Analysetechnik kann eine sehr komfortable und effektive Programmablaufverfolgung durchgeführt werden, da unerwünschte Informationen, z. B. LDIR-Befehle, nicht mit aufgezeichnet werden brauchen. Weiterhin besteht die Möglichkeit, verschiedene Analysemodi in einem Lauf gemeinsam durchzuführen, also beispielsweise einzelne Datenproben mit dem DEBUG-Kommando an ausgewählten Befehlen zu entnehmen und gleichzeitig verschiedene interessierende Programmausschnitte aufzuzeichnen. Programme mit komplizierten Interruptstrukturen können mit Hilfe solch kombinierter Aufzeichnungen, in die auch Interruptvektoren bzw. gesamte Interruptserviceroutinen einbezogen werden können, komfortabel analysiert werden.

Serviceanalyse

Ein weiteress breites Anwendungsfeld liegt im Servicebereich. Steuerungen mit Mikrorechnerminimalkonfigurationen, die oft selbst über keinen eigenen Maschinenmonitor verfügen, können effektiv bei auftretenden Fehlfunktionen analysiert werden. In diesem Zusammenhang soll eine weitere Betriebsart des autonomen Softwareanalysegerätes erläutert werden. Diese ermöglicht das Auffinden der Ursache von Fehlern, die zum Programmabsturz führen und, wenn sie selten auftreten, oftmals nur schwer zu finden sind. Im sogenannten SET-ADR-COUNTER-Mode wird der gesamte Adreßbereich des zu testenden Programmsystems mit dem oben erwähnten Kommando markiert. Dabei sind Blöcke mit Zwischenräumen zulässig. Während des Echtzeitlaufes wird der Programmablauf ständig zyklisch abgespeichert (END TRACE). Sobald das Programm aufgrund eines Fehlers aus dem markierten Bereich gerät, wird die Aufzeichnung abgebrochen. Für die Fehleranalyse stehen dann 1023 der letzten Maschinenzyklen des untersuchten Programms zur Auswertung mit dem Disassembler zur Verfügung. In den meisten Fällen kann aus der Vorgeschichte sehr schnell auf die Fehlerursache geschlossen werden.

Ereigniszählung

In dieser Betriebsart kann die Häufigkeit des Durchlaufs bestimmter Programme festgestellt werden.

Zeitmessung

Die Zeit zwischen zwei Programmstellen kann mit einer Auflösung von 1 µs und einem Eindeutigkeitsbereich von 13 Sekunden bestimmt werden. Die Zeitmessung ist besonders wertvoll in Fällen, in denen die Laufzeit nicht nur von der Summe der Befehlslaufzeiten, sondern in Verbindung mit einem Interface von einer äußeren Quelle mitbestimmt wird.

Unterbrechung der Aufzeichnung (Break)

Die Aufzeichnung kann beim Erreichen vorgewählter Adressen unterbrochen werden.
Das im EPROM gespeicherte Betriebssystem von 12 KByte Länge ermöglicht eine komfortable Bedienung und zahlreiche Auswertemöglichkeiten, die eine hohe Effektivität der Softwareanalyse gewährleisten.

Nachnutzung

Das autonome U880-Softwareanalysegerät kann über das BfN des VEB Mikroelektronik „Karl Liebknecht" Stahnsdorf, Ruhlsdorfer Weg, Stahnsdorf, 1533, nachgenutzt werden. Gegenstand der Nachnutzung sind eine vollständige und ausführliche Hard- und Software-Gerätedokumentation und die Bereitstellung eines unbestückten Lichterplattensatzes. Bei der Bestückung der Leiterplatten sowie der Inbetriebnahme des Gerätes kann Hilfestellung nach individueller Absprache gewährt werden.

✉ KONTAKT ☏

Ingenieurbetrieb für die Anwendung der Mikroelektronik Potsdam, Leipziger Straße 57, Potsdam, 1560; Tel. 2 21 95

Schaltplanerstellung auf dem KC 85/2

Das im folgenden beschriebene Programm *Hydraulik für KC 85/2* wurde im Rahmen einer Aufgabe *Erarbeitung der notwendigen Grundsoftware zur bildschirmgestützten Entwicklung von Funktionsschaltplänen mittels KC 85/2* entwickelt.
Das Programm beinhaltet die grafische Darstellung von Hydraulik-Funktions- und Bauschaltplänen sowie die Druckerroutinen für einen grafikfähigen Epson-Nadeldrucker. Für andere grafikfähige Drucker ist es leicht möglich, die dafür notwendige Druckerroutine zu installieren.

Hardwaremäßige Voraussetzungen:

- KC 85/2 mit RAM-Erweiterung Modul 022 oder 011
- V.24-Schnittstelle sowie grafikfähiger Drucker
- Kassettenmagnetbandgerät
- Farbmonitor mit RGB-Anschluß bzw. schwarz-weiß-Monitor mit BAS-Anschluß.

Die Verwendung von RGB- bzw. BAS-Anschluß ist unbedingt erforderlich, um die Zeichen und Symbole deutlich erkennen zu können. Die Bildwiedergabequalität über Video-Anschluß genügt den Anforderungen hierbei nicht. Das Programm ist in Assemblersprache geschrieben. Damit wird eine höhere Verarbeitungsgeschwindigkeit erreicht und die begrenzte Speicherkapazität optimal genutzt. Auch kann eine maximale Anzahl von Symbolen und deren logische Ausgabe auf das Display bzw. den Drucker mit der begrenzten Speicherkapazität des KC 85/2 realisiert werden. In dem Hydraulikprogramm

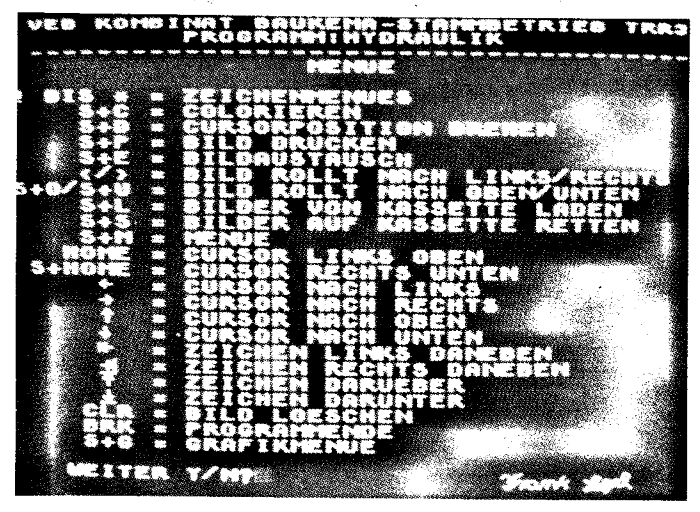



Bild 1 Erläuterungsmenü

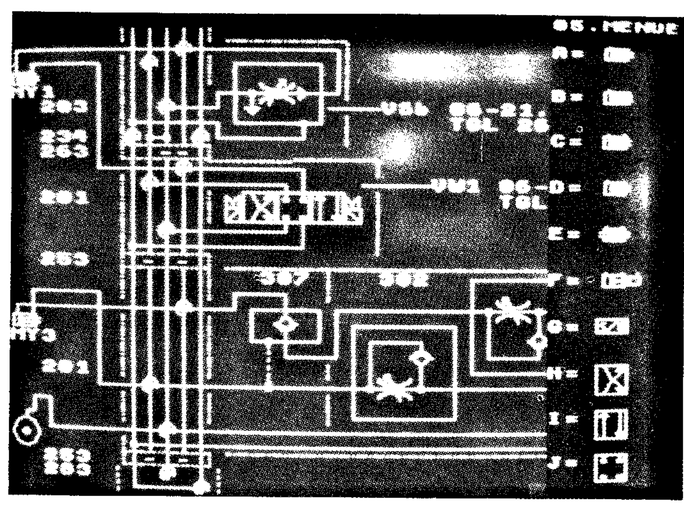
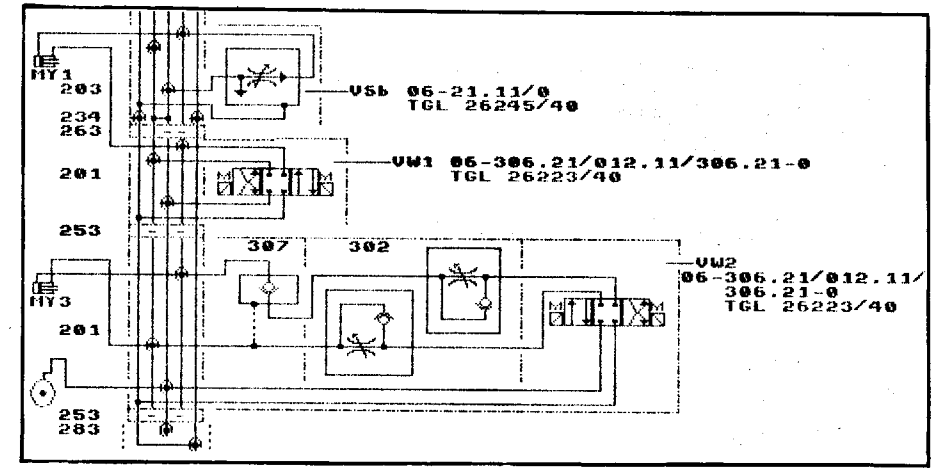


Bild 2 Funktionsschaltplan

Bild 3 Zwei nebeneinander auf LX-86 ausgedruckte Bilder *Fotos: Clauss (3)*



sind die wichtigsten Zeichen und Symbole laut TGL 8672 enthalten; die Auswahl erfolgte nach betrieblichen Erfordernissen. Andere Zeichen und Symbole lassen sich relativ einfach programmieren. Das Programm unterteilt sich in ein Hauptmenü zur Beschreibung der Tastenfunktion, wobei S die Umschalttaste (-shift) und gleichzeitig die Betätigung einer weiteren Taste bedeutet, und in den Zeichenmodus.

Im Zeichenmodus wird der Bildschirm in ein Zeichenfenster mit 32 Zeilen und 32 Spalten und in ein Menüfenster von 32 Zeilen und 8 Spalten aufgeteilt.

In dem Menüfenster lassen sich 9 verschiedene Grafikmenüs aufrufen, die teilweise aus Zeichen (eine Cursorgröße und teilweise aus Symbolen bis 6 Cursorgrößen) bestehen. Die Zeichen und Symbole lassen sich mit den jeweils dazugehörenden Buchstaben aufrufen und werden im Zeichenfenster auf der Cursorposition bzw. logisch fortlaufend abgebildet. Wenn mehrere gleiche Zeichen nacheinander verwendet werden, wie Ziehen von Leitungsverbindungen, läßt sich dies leicht realisieren durch Betätigung der Shifttaste und gleichzeitig einer Cursorbewegungstaste. Dadurch wird automatisch das zuletzt eingegebene Zeichen auf der neuen Cursorposition ausgegeben. Bei längerer Betätigung beider Tasten springt der Cursor in die Richtung der Cursorbewegungstaste weiter, wobei immer das zuletzt eingegebene Zeichen ausgegeben wird. Diese Art der Zeichnungserstellung ist aus der Fachliteratur als semigrafische Darstellung bekannt.

Für die 9 Grafikmenüs sind verschiedene Zeichengeneratoren mit jeweils 26 Zeichen (von A-Z) verwendet worden.

Durch die begrenzte Speicherkapazität ist es nur möglich, zwei Bilder des Zeichenfensters pixelweise abzuspeichern.

Der Speicherplatz kann z. B. mittels Kopplung von zwei KC 85/2 über die V.24-Schnittstelle erweitert werden. Damit können entsprechend den Anforderungen mehrere Bilder pixelweise gespeichert werden.

Da diese Bilder außerhalb des Bildwiederholspeichers abgelegt sind, ergibt sich die Möglichkeit, in das Hauptmenü zu schalten und nach Rückkehr in den Zeichenmodus nach automatischem Erscheinen des letzten aktuellen Bildes weiterzuzeichnen.

Wie bei dem Programm *Elektro* (zur Erstellung von elektrischen und elektronischen Schaltplänen) können die beiden Bilder durch Rollen nach oben/unten und links/rechts ineinander übergehen bzw. direkt ausgetauscht werden.

Eine weitere Variante der Bildmanipulation ist die Verwendung von INS und DEL. Hierbei wird ab der Stelle des Cursors das ganze Bild senkrecht verändert. Dadurch können mit INS Spalten eingefügt bzw. mit DEL gelöscht werden.

Für eine übersichtliche Darstellung bzw. Kontrolle der Schaltpläne kann man mit einem Farbmonitor ein Bild colorieren. Das heißt, nach vorheriger Farbauswahl im Coloriermodus lassen sich die hervorzuhebenden Stellen farbig gestalten, indem man den Cursor über diese Stellen bewegt. Dadurch läßt sich z. B. der Ölstrom zum Verbraucher verdeutlichen.

Diese Farbgestaltung ist nur auf ein Bild begrenzt. Eine zusätzliche Einschränkung besteht darin, daß dabei nicht gerollt (oben/unten, links/rechts) werden kann, weil die Farbinformationen nur im Color-RAM des Bildwiederholspeichers gespeichert werden. Eine weitere Speicherung der Farbinformation ist aus Gründen der begrenzten Speicherkapazität des KC 85/2 nicht gegeben. Deshalb ist diese Farbgestaltung für die Weiterbearbeitung der Schaltpläne löschbar.

Der Einsatz der Vollgrafik (256 × 320 Bildpunkte) ist im Grafikmenü S + G (Shift und G) anwendbar. Es können beliebige Punkte mittels Cursorbewegungstasten gesetzt bzw. gelöscht werden. Das Punktsetzen wird im Grafikmenü mittels S + Z (Shift und Zeichen), das Punktlöschen mittels S + L (Shift und Löschen) eingestellt. Der Beginn des Zeichens ist mit der letzten Cursorposition identisch. Für das Ausdrucken der auf dem Monitor entwickelten und pixelweise gespeicherten Bilder ergeben sich nachfolgende Varianten, die beim Aufbau der Bilder bereits zu berücksichtigen sind.

1. Zwei Bilder 90° gedreht nebeneinander
2. Zwei Bilder nebeneinander
3. ein Bild 90° gedreht.

Für das Zusammensetzen eines einzelnen Druckbildes aus zwei im Arbeitsspeicher des Rechners abgelegten Bildern ergeben sich folgende Kombinationen:

- untereinander anschließend
- nebeneinander anschließend
- einzeln.

Bei der Entwicklung von Funktionsschaltplänen stellt diese Art der Druckbildgestaltung eine wesentliche Erweiterung dar.

Es ist zu beachten, daß das auf dem Monitor befindliche Bild nach Druckaufruf als erstes Bild gedruckt wird.

Bei der 3. Druckvariante ist nach Ausdruck des ersten Bildes mittels S + E (Shift und E) ein Bildaustausch vorgesehen, und nach erneutem Druckaufruf kann das 2. Bild ausgedruckt werden.

Obwohl die Befehle zur Bildteilemanupulation begrenzt sind (nur INS + DEL vertikal über das gesamte Bild) sowie Einfügen und Löschen von Zeilen nicht ausführbar ist, hat sich dieses Programm bei uns in der Praxis bewährt und zeigt weitere Einsatzmöglichkeiten des Kleincomputers KC 85/2 auf.

Frank Legel

⊠ KONTAKT ☏

VEB Kombinat baukema, Stammbetrieb, Katharinenstr. 17, Abt. TRR 3, Leipzig, 7010; Tel. Rötha 3 67 54

Programmieren mit MACRO–SM

Teil IV
Dr. Thomas Horn
Informatikzentrum des Hochschulwesens an der Technischen Universität Dresden

5. *Der Befehlssatz*

In diesem Abschnitt werden die wesentlichen Kenntnisse über den Befehlssatz der Rechenanlagen des SKR für den Programmierer in MACRO-SM zusammengefaßt. Das Format der symbolischen Maschinenbefehle entspricht dem gemäß Punkt 1.1. dargelegten allgemeinen Format von Assembleranweisungen. Als Voraussetzung für die Anwendung der Maschinenbefehle werden zunächst die Registerstruktur, die Befehlsformate und die Adressierungsarten betrachtet.

5.1. Die Registerstruktur der Prozessoren

Dem Programmierer steht ein Registerblock mit 8 allgemeinen Mehrzweckregistern und das Prozessorstatusregister zur Verfügung. Die 8 allgemeinen Register können direkt durch die Befehle adressiert werden. Sie werden mit einer 3-Bit-Nummer kodiert. Man nennt sie deshalb gewöhnlich Register 0 (**R0**), Register 1 (**R1**), ..., Register 7 (**R7**). Diese Register können als Akkumulatoren, Adressierungsregister, Indexregister, Kellerzeiger (Stackpointer) und Zwischenregister zur Speicherung von Daten und Adressen benutzt werden.
Die Register haben eine Länge von 16 Bit (1 Wort). Von den Byteverarbeitungsbefehlen werden die niederwertigen 8 Bit der Register beeinflußt. Eine Ausnahme bildet nur das Laden eines Bytes, wobei die höherwertigen 8 Bits des Registers auf 0 oder 1 gesetzt werden, entsprechend dem Vorzeichen des zu ladenden Bytes. Dadurch erfolgt die automatische Umwandlung von Daten aus dem Byte- in das Wortformat.
Die Register R6 und R7 nehmen unter den allgemeinen Registern eine Sonderstellung ein. Das Register R7 wird von der Gerätetechnik als Befehlszähler (**PC** – Program counter) benutzt. Der **PC** enthält immer die Adresse des nächsten abzuarbeitenden Befehls. Daraus ergeben sich einige Einschränkungen für die Benutzung des Registers R7, da nur einige Operationen sinnvoll sind. Zum Beispiel ist das Laden der Konstante 1000 in das Register R7 identisch mit einem Sprung zur Adresse 1000; dagegen ist das Bilden des Komplements vom Inhalt des Registers R7 eine absurde Operation, die zu einer Programmausnahme führt.
Das Register R6 wird als Stackpointer (**SP**) für einen sogenannten Systemstack benutzt. Dieser Systemstack ist für die Unterprogramm- und Interruptorganisation notwendig. Der Systemstack wird vom Betriebssystem durch Laden der Adresse eines speziell dafür reservierten Hauptspeicherbereiches in den **SP** eingerichtet. Bei einem Unterprogrammruf wird in den Systemstack die Rücksprungadresse zur späteren Fortsetzung des Hauptprogramms eingetragen. Bei einem Interrupt wird der Inhalt des **PC** und des Prozessorstatusregisters in den Systemstack gerettet, so daß nach der Interruptbehandlung das unterbrochene Programm fortgesetzt werden kann. Da der Systemstack einen allgemeinen temporären Arbeitsspeicherbereich darstellt, wird von ihm außerdem bei der Programmierung zum Retten von Registerinhalten, zur Parameterübermittlung bei der Unterprogrammorganisation usw. Gebrauch gemacht.
Das Prozessorstatusregister (**PS**) ist ein 16-Bit-Register, das die Bedingungsflags und die Priorität des laufenden Programms beinhaltet. Auf das **PS** kann nur über seine Busadresse 777776(8) zugegriffen werden. Eine Ausnahme sind der K 1620 und die Elektronika-60, bei denen der Zugriff auf das **PS** über 2 spezielle Befehle (**MTPS, MFPS**) erfolgt. Die Bedingungsflags werden von der arithmetisch-logischen Einheit gesetzt und von den bedingten Verzweigungsbefehlen ausgewertet. Die Programmpriorität wird vom Programm oder Betriebssystem gesetzt und von der Interruptsteuerung benutzt.

5.2. Der Befehlsaufbau

Ein Befehl belegt in seiner Grundstruktur 16 Bit. Bei verschiedenen Adressierungsmodifikationen wird der Befehl durch Direktwerte, relative und/oder absolute Adressen auf zwei oder drei Worte erweitert. Die Befehle können in verschiedene Befehlsgruppen unterteilt werden:

1. Ein-Adreß-Befehle
2. Zwei-Adreß-Befehle
3. Verzweigungsbefehle
4. Spezialbefehle (adressenlose Befehle).

Die Befehle bestehen im allgemeinen aus einem Operations- und einem Operandenteil. Da ein Befehl in der Grundstruktur nur 16 Bit belegt, ist die Angabe von echten Hauptspeicheradressen nicht möglich. Deshalb erfolgt die Operandenadressierung bei den Ein- und Zwei-Adreß-Befehlen über die allgemeinen Register. Zusätzlich zur Registerangabe wird bei den Befehlsstrukturen ein Adressierungsmodus spezifiziert, der die Art und Weise der Verwendung des Registers angibt. Die sich daraus ergebenden verschiedenen Adressierungsarten werden nachfolgend detailliert betrachtet.
Die meisten Ein- und Zwei-Adreß-Befehle sind für die Verarbeitung von Operanden im Wort- und Byteformat realisiert (Wort- und Byteverarbeitungsbefehle). Zur Unterscheidung dieser beiden Befehlstypen wird das Bit 15 des Befehlswortes benutzt:
Bit 15 = 0 – Wortverarbeitungsbefehl
Bit 15 = 1 – Byteverarbeitungsbefehl.

5.2.1. Ein-Adreß-Befehle

Der Aufbau der Ein-Adreß-Befehle ist in Bild 2 dargestellt. Die meisten Ein-Adreß-Befehle sind vom Typ 1 und enthalten einen Operationskode (10 Bit) und die Beschreibung der Zieladresse, bestehend aus dem Modus (3 Bit) und der Registernummer (3 Bit).
Dagegen sind nur wenige Ein-Adreß-Befehle vom Typ 2. Dazu zählt der Befehl „Rücksprung aus dem Unterprogramm".
Bei diesen Befehlen wird der Modus der Verwendung des Registers durch den Operationskode festgelegt.
Beispiel:
Löschen des Registers 3
Bei diesem Befehl steht der Operand direkt im allgemeinen Register. Die direkte Registeradressierung ist der Modus 0. In dualer Schreibweise:

0 000 101 000	000	011
Operationskode	Modus	Register

In oktaler Schreibweise: **005003**
In symbolischer Schreibweise: **CLR R3**

5.2.2. Zwei-Adreß-Befehle

Der Aufbau der Zwei-Adreß-Befehle ist in Bild 3 dargestellt. Die meisten Zwei-Adreß-Befehle sind vom Typ 1. Der Operationskode umfaßt 4 Bit und die beiden Adreßangaben, Quell- und Zieladresse, jeweils 6 Bit. Sie bestehen aus einer Register- und Modusangabe.
Es gibt nur wenige Zwei-Adreß-Befehle vom

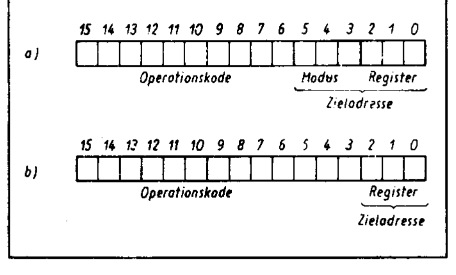
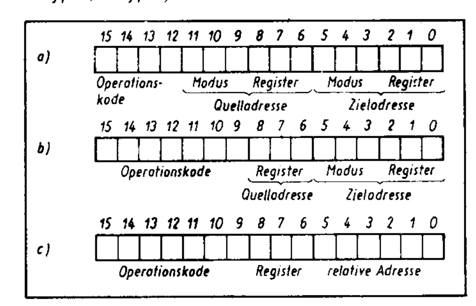


Bild 2 Aufbau der Ein-Adreß-Befehle (a-Typ 1, b-Typ 2)

Bild 3 Aufbau der Zwei-Adreß-Befehle (a-Typ 1, b-Typ 2, c-Typ 3)



Typ 2. Dazu zählen der „Unterprogrammsprungbefehl“ und die Befehle des erweiterten Befehlssatzes. Das Register für die Quelloperandenangabe wird dem Modus 0 entsprechend verwendet (Operand im Register). Es gibt nur einen Zwei-Adreß-Befehl vom Typ 3, den Zyklus-Befehl. Das Register wird zur Zählung der Schleifen benutzt, und die relative Adresse stellt eine Sprungdistanz (in Worten!) zum Schleifenanfang dar. Die relative Adresse wird somit doppelt vom aktuellen Befehlszählerstand subtrahiert. Es ist zu beachten, daß dabei der Befehlszähler (PC) schon auf den nächsten Befehl zeigt.

Beispiel:
Transport eines Operanden aus Register 5 in das Register 3. Der Adressierungsmodus ist auf Grund der direkten Registeradressierung für beide Adressen 0.
In dualer Schreibweise:

0 001	000	101	000	011
Operations-kode	Modus	Reg.	Modus	Reg.
	Quelladr.		Zieladresse	

In oktaler Schreibweise: **010503**
In symbolischer Schreibweise: **MOV R5, R3**

5.2.3. *Verzweigungsbefehle*

Der Aufbau der Verzweigungsbefehle ist in Bild 4 dargestellt: Im höherwertigen Byte steht der Operationskode und im niederwertigen Byte eine Verschiebung (relative Adresse), die eine Sprungdistanz (in Worten!) im Zweierkomplement zum Sprungziel darstellt. Addiert man die Verschiebung doppelt zum aktuellen Befehlszählerstand, so erhält man die Adresse des Sprungzieles. Es ist darauf zu achten, daß der Befehlszähler dabei schon auf den nächsten Befehl zeigt.

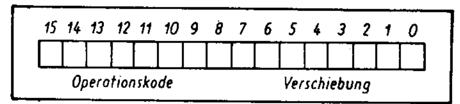


Bild 4 Aufbau der Verzweigungsbefehle

5.2.4. *Spezialbefehle*

Unter diese Kategorie fallen alle adressenlosen Befehle, wie:
- Steuerbefehle (**HALT, WAIT** usw.)
- Setzen/Löschen der Bedingungsflags (SEC, CLC, ...)
- Trap-Befehle (Unterbrechungsbefehle).

Bei den Bedingungsflag- und Steuerbefehlen belegt der Operationskode alle 16 Bit des Befehlswortes. Beim **TRAP-** und **EMT-**Befehl enthält das niederwertige Byte die Nummer des **TRAP-** oder **EMT-**Befehls.

5.3. *Adressierungsarten*

Die Adressierungsart wird durch den Adressierungsmodus bestimmt, der im Befehl in der Regel für jeden Operanden mit 3 Bit kodiert wird. Daraus ergeben sich die nachfolgend beschriebenen 8 Grundadressierungsarten, die gerätetechnisch realisiert sind. Als erweiterte Adressierungsart werden die Adressierungsarten bezeichnet, die sich programmtechnisch unter Anwendung des Befehlszählers (**PC**) als Adressierungsregister und einiger Grundadressierungsarten organisieren lassen (auch **PC**-Adressierungsmodifikationen genannt). Sie werden programmtechnisch durch den Assembler unterstützt, der bei der Übersetzung der speziellen Notationen den entsprechenden Maschinenkode erzeugt, der sich auf die beschriebenen Grundadressierungsmodifikationen zurückführen läßt. In den Assemblernotationen werden folgende Kurzbezeichnungen verwendet:

E – ist ein Ausdruck gemäß 3.6, der eine 16-Bit-Adresse ergibt, die entsprechend dem Charakter des PA absolut bzw. absolut oder relativ sein kann, z. B. **ALPHA+2.**

R – ist ein Registerausdruck, der entweder das Registerkennzeichen % oder ein Registersymbol enthält, z. B. **PC, R0, %REG+3, %10/2, R2+1, %2+1.**

ER – ist ein Registerausdruck oder Ausdruck, der einen Wert von 0 bis 7 ergibt.

Die Verwendung eines Ausdrucks zur Bezeichnung eines Registers ist nur dann eindeutig, wenn der Ausdruck in runde Klammern eingeschlossen ist, die der Bedeutung „Register-Indirekt“ entsprechen.

Eine zusammenfassende Übersicht über die Adressierungsarten ist in Tafel 9 enthalten.

5.3.1. *Die Grundadressierungsarten*

Registeradressierung (Modus 0)

Bei der Registeradressierung (Bild 5) wird der Registerinhalt direkt als Operand benutzt. Da die allgemeinen Register integrierter Bestandteil des Prozessors sind, ergibt diese Adressierungsart die kürzeste Befehlsausführungszeit. Für den Zugriff zum Operanden wird kein Buszyklus benötigt.
Das Register *R* enthält den Quell- oder Zieloperanden.

Beispiele:
```
INC R2; ERHOEHEN DES
;REGISTERINHALTES UM 1
CLR %2; LOESCHEN DES REGISTERS %2
```

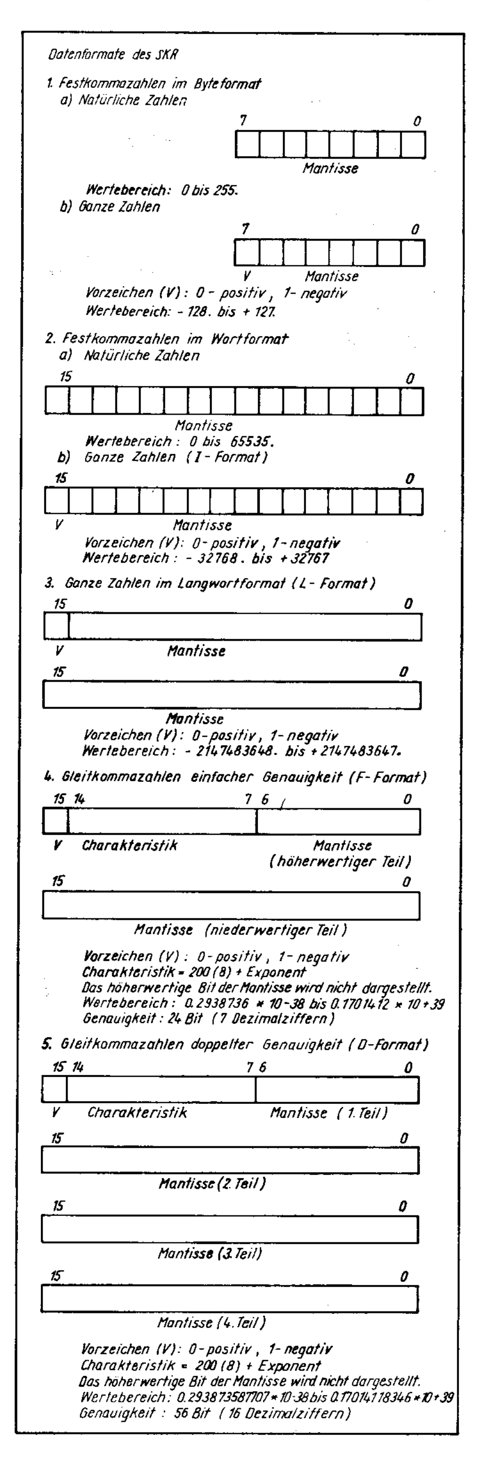
Register-Indirekt-Adressierung (Modus 1)

Bei der Register-Indirekt-Adressierung (Bild 6) wird der Inhalt des adressierten Registers als die Adresse des Operanden interpretiert. Für den Zugriff zum Operanden wird ein Buszyklus benötigt.
Das Register enthält die Adresse des Operanden im Hauptspeicher. Der Assembler läßt auch die Notation @*R* zu, wobei aber nur ein Registerausdruck *R* zulässig ist.

Beispiele:
```
INC (2) ; ADRESSE IN REGISTER 2
CLR @%2
CLR @R2
CLR (R2)
```

Autoinkrement-Adressierung (Modus 2)

Bei der Autoinkrement-Adressierung (Bild 7) wird wie bei der Register-Indirekt-Adressierung der Inhalt des adressierten Registers als die Adresse des Operanden interpretiert.
Für den Zugriff zum Operanden wird ein Buszyklus benötigt. Anschließend wird der Registerinhalt um die Länge des Operanden erhöht, bei Byteverarbeitungsbefehlen um 1 und bei Wortverarbeitungsbefehlen um 2.
Bei den Registern 6 und 7 erfolgt immer eine Erhöhung um 2.
Das Register *ER* enthält die Adresse des



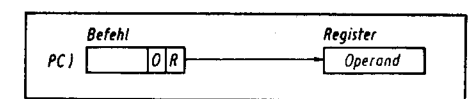


Bild 5 Registeradressierung

Operanden. Das nachgestellte Pluszeichen (+) zeigt an, daß die Adresse im Register nach der Operation erhöht wird.

Beispiele:
INC (2)+
CLR (R2)+

Autoinkrement-Indirekt-Adressierung (Modus 3)

Bei der Autoinkrement-Indirekt-Adressierung (Bild 8) wird der Inhalt des adressierten Registers als eine Adresse einer im Speicher befindlichen Adreßkonstanten der Operandenadresse interpretiert. Damit sind für den Operandenzugriff zwei Buszyklen erforderlich. Nach dem Zugriff zum Operanden wird der Inhalt des Registers immer um 2 erhöht, da das Register auf eine Adreßkonstante (2 Bytes) zeigt.
Das Register *ER* enthält die Adresse (Register-Indirekt) der Adresse (Speicher-Indirekt) des Operanden. Nach der Operation wird die Adresse im Register erhöht.

Beispiele:
INC @(2)+
CLR @(R1+1)+

Autodekrement-Adressierung (Modus 4)

Bei der Autodekrement-Adressierung (Bild 9) wird der Inhalt des adressierten Registers um die Länge des Operanden verringert (bei Byteverarbeitungsbefehlen um 1, bei Wortverarbeitungsbefehlen um 2). Bei den Registern 6 und 7 erfolgt immer eine Verringerung um 2. Anschließend wird der verringerte Inhalt des Registers als Adresse des Operanden interpretiert. Für den Zugriff zum Operanden wird ein Buszyklus benötigt. Das Registers *ER* enthält die Adresse des Operanden. Das vorgestellte Minuszeichen (−) zeigt an, daß die Adresse im Register vor der Operation erniedrigt wird.

Beispiele:
INC −(R2)
CLR −(%1+2)

Autodekrement-Indirekt-Adressierung (Modus 5)

Bei der Autodekrement-Indirekt-Adressierung (Bild 10) wird der Inhalt des adressierten Registers um 2 verringert. Anschließend wird der verringerte Inhalt des Registers als Adresse einer Adreßkonstanten der Adresse des Operanden interpretiert. Für den Zugriff zum Operanden werden zwei Buszyklen benötigt.
Das Register *ER* enthält die Adresse (Register-Indirekt) der Adresse (Speicher-Indirekt) des Operanden. Vor der Operation wird die Adresse im Register erniedrigt.

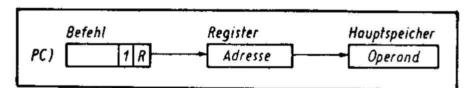


Bild 6 Register-Indirekt-Adressierung

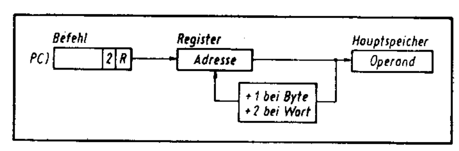


Bild 7 Autoinkrement-Adressierung

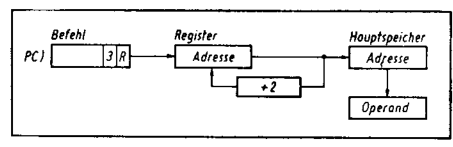


Bild 8 Autoinkrement-Indirekt-Adressierung

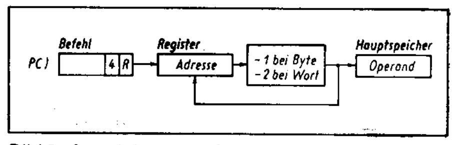


Bild 9 Autodekrement-Adressierung

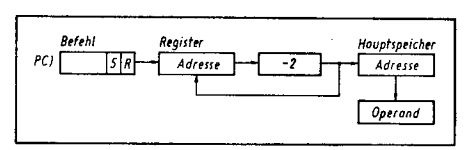


Bild 10 Autodekrement-Indirekt-Adressierung

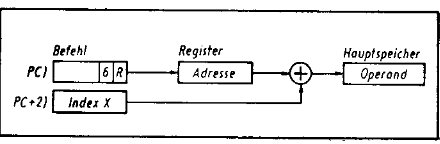


Bild 11 Index-Adressierung

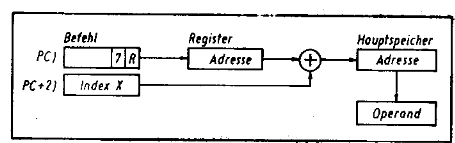


Bild 12 Index-Indirekt-Adressierung

Beispiele:
INC @−(R2)
CLR @−(2+1)

Index-Adressierung (Modus 6)

Bei der Index-Adressierung (Bild 11) wird zum Inhalt des adressierten Registers eine Indexkonstante addiert. Die Summe wird als die Adresse des Operanden interpretiert.
Die Indexkonstante belegt das nächste Wort nach dem Befehl. Der Befehlszähler (**PC**) wird nach dem Lesen der Indexkonstanten um 2 erhöht. Für den Zugriff zum Operanden werden zwei Buszyklen benötigt.
Der Ausdruck *E*, mit dem Inhalt des Registers *ER* addiert, ergibt die Adresse des Operanden. Der Wert des Ausdrucks *E* wird nach dem Befehlswort gespeichert. Der Ausdruck kann programmtechnisch die Basisadresse oder die Indexkonstante spezifizieren.

Beispiele:

```
INC FELD(R2); IM BEREICH FELD WIRD
            ; DAS DURCH R2 SPEZIFI-
            ; ZIERTE
            ; WORT INKREMENTIERT

CLR 20(R3) ; LOESCHEN DES WORTES
           ; MIT DER VERSCHIEBUNG 20
           ; IM BEREICH, DER DURCH
           ; R3 SPEZIFIZIERT WIRD.
```

Index-Indirekt-Adressierung (Modus 7)

Bei der Index-Indirekt-Adressierung (Bild 12) wird der Inhalt des adressierten Registers mit einer Indexkonstanten addiert. Die Summe wird als die Adresse einer Adreßkonstanten der Adresse des Operanden interpretiert. Die Indexkonstante belegt das nächste Wort nach dem Befehl. Nach dem Lesen der Indexkonstanten wird der Befehlszähler (**PC**) um 2 erhöht. Für den Zugriff zum Operanden werden drei Buszyklen benötigt. Der Ausdruck *E*, mit dem Inhalt des Registers *ER* addiert, ergibt die Adresse der Adresse des Operanden. Der Wert des Ausdrucks *E* wird nach dem Befehlswort gespeichert.

Beispiele:
INC @FELD(R2)
CLR @10.(R1+2)

Anmerkungen zu Byteverarbeitungsbefehlen:

1. Bei den Adressierungsmodifikationen mit Inkrementierung bzw. Dekrementierung des Registerinhaltes (2 und 4) erfolgt bei den Byteverarbeitungsbefehlen eine Erhöhung bzw. Erniedrigung der Adresse um 1 und bei den Wortverarbeitungsbefehlen um 2.
2. Bei den indirekten Adressierungsmodifikationen mit Inkrementierung bzw. Dekrementierung des Registerinhaltes (3 und 5) erfolgt bei den Byte- und Wortverarbeitungsbefehlen eine Erhöhung bzw. Erniedrigung der Adresse um 2, da die Register auf die Adressen der Operanden verweisen.
3. Byteverarbeitungsbefehle in Verbindung mit dem Stackpointer des Systemstacks (**SP**) und dem Befehlszähler (**PC**) erhöhen bzw. erniedrigen diese Register immer um 2.
4. Byteverarbeitungsbefehle adressieren bei der Registeradressierung das niederwertige Byte im Register. Bei Byteverarbeitungsbefehlen in Verbindung mit dem **SP** bzw. **PC** wird ebenfalls immer das niederwertige Byte eines Wortes adressiert.
5. Beim Laden eines Bytes in ein Register (**MOVB**) wird das Byte in den niederwertigen Teil des Registers geladen, und der höherwertige Teil des Registers wird mit dem Vorzeichen des Bytes gefüllt (Signextension). Damit wird eine ganze Zahl aus dem Byteformat in das Wortformat überführt.

5.3.2. Die erweiterten Adressierungsarten
Direktwertadressierung (Modus 2 mit PC)

Der Wert des Operanden wird als sogenannter Direktwert (Bild 13) nach dem Befehlswort gespeichert. Da nach dem Lesen des Be-

fehlswortes der **PC** auf das nächste Wort zeigt, kann unter Verwendung des **PC** als Adressierungsregister der Direktwert gelesen werden. Bei Verwendung des Modus 2 wird der **PC** automatisch um 2 erhöht, so daß er nach Ausführung des Befehls mit Direktwertadressierung auf den nächsten Befehl zeigt.
Die Direktwertadressierung wird vor allem bei Zwei-Adreß-Befehlen für die Angabe des Quelloperanden verwendet, wenn er z. B. eine Konstante oder Maske darstellt. Der Assembler erkennt Direktwerte am vorangestellten Doppelkreuz „#".
Der Ausdruck *E* ist der Operand selbst. Der Wert des Ausdrucks wird nach dem Befehlswort als Direktwert gespeichert.

Beispiele:

```
MOV #FELD,R2    ; DIE ADRESSE FELD
                ; WIRD GELADEN
ADD #^B1011,R2  ; ZUM INHALT DES
                ; REGISTERS R1
                ; WIRD DER WERT
                ; 1011(2) ADDIERT
```

Absolute Adressierung (Modus 3 mit PC)

Die absolute Adresse des Operanden wird als sogenannter Direktwert (Bild 14) nach dem Befehlswort gespeichert. Da nach dem Lesen des Befehlswortes der **PC** auf den Direktwert zeigt, kann unter Verwendung des **PC** als Adressierungsregister und des Modus 3 zum Operanden zugegriffen werden. Dabei wird gleichzeitig der **PC** um 2 erhöht, so daß der **PC** nach Ausführung des Befehls auf den nächsten Befehl zeigt. Die absolute Adressierung soll nur dort eingesetzt werden, wo die Adresse absolut feststeht, z. B. zur Adressierung des Prozessorstatuswortes, der Geräteregister und Interruptvektoren. Der Assembler erkennt die absolute Adressierung an dem vorangestellten Sonderzeichen „@#".
Der Ausdruck *E* gibt die Adresse des Operanden an, die nach dem Befehlswort gespeichert wird. Der Ausdruck darf je nach Charakter des PA absolut oder relativ sein. Bei relativen (verschieblichen) Adressen erfolgt vom Taskbuilder die Umrechnung in eine absolute Adresse. Zur Programmlaufzeit steht somit hinter dem Befehlswort immer die absolute Adresse.

Beispiele:

```
MOV @#177776,R0; LADEN
               ; DES PROZESSOR-
               ; STATUS
CLR @#ALPHA
```

Relative Adressierung (Modus 6 mit PC)

Da sich bei der Index-Adressierung die Operandenadresse aus der Summe des Inhaltes des Adressierungsregisters und der Indexkonstanten ergibt, muß bei Verwendung des **PC** als Adressierungsregister die Indexkonstante die Differenz zwischen Operandenadresse und dem aktuellen Inhalt des **PC** betragen (Bild 15). Diese Differenz wird als relative Adresse bezeichnet. Diese Adressierungsart ist die am meisten verwendete. Der Assembler übersetzt alle Operandenangaben ohne besonderes Kennzeichen als relative Adressen.
Der Ausdruck *E* gibt je nach Charakter des PA die absolute oder relative (verschiebliche) Adresse des Operanden an. Als relative Adresse im Sinne der relativen Adressierung wird die Differenz zwischen dem Wert des Ausdrucks und der Adresse **PC** nach dem Lesen der Indexkonstanten des Befehls berechnet und als Indexkonstante nach dem Befehlswort gespeichert. Ist die Berechnung der relativen Adresse zur Assemblerzeit nicht möglich, so wird sie vom Taskbuilder ausgeführt. Zur Programmlaufzeit steht somit immer die relative Adresse des Operanden zum aktuellen **PC**-Inhalt nach dem Lesen der relativen Adresse im Wort der Indexkonstanten. Diese Adressierungsart ist für die Adressierung von symbolischen Speicherbereichen die am meisten genutzte.

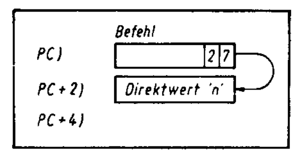


Bild 13 Direktwertadressierung

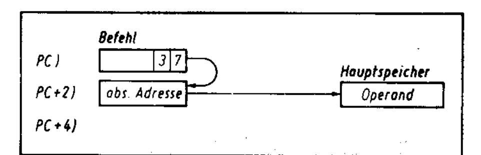


Bild 14 Absolute Adressierung

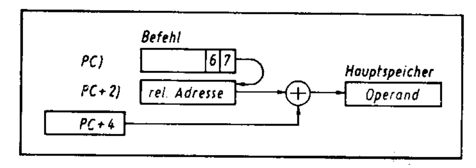


Bild 15 Relative Adressierung

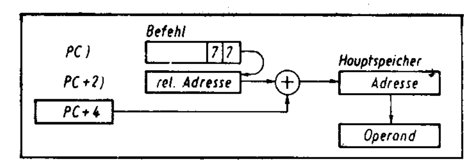


Bild 16 Indirekte relative Adressierung

Beispiele:

```
MOV ALPHA, BETA
CLR A+B-C
```

Indirekte relative Adressierung (Modus 7 mit PC)

Aus der Summe der relativen Adresse, die nach dem Befehlswort gespeichert ist, und dem Inhalt des PC ergibt sich die absolute Adresse einer Adreßkonstanten des Operanden (Bild 16). Der Assembler erkennt diese Adressierungsart am vorangestellten Sonderzeichen „@".
Der Ausdruck *E* gibt je nach Charakter des PA die absolute oder relative (verschiebliche) Adresse der Adresse des Operanden an. Weitere Bemerkungen siehe unter „Relative Adressierung".

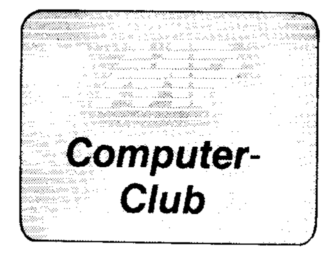
Beispiele:

```
MOV @ADR, @BETA
CLR @A+2*<C-D>
```

5.3.3. Relative Adressierung in den Verzweigungsbefehlen

Der unbedingte und die bedingten Verzweigungsbefehle sind 1-Wort-Befehle, die im niederwertigen Byte eine relative Adresse (Verschiebung) enthalten. Diese Verschiebung (*V*) gibt die Anzahl der Worte an, um die vorwärts (+) oder rückwärts (−) gesprungen werden soll. Die Ausdrucksberechnung erfolgt wie bei der relativen Adressierung – mit dem Unterschied, daß anschließend der Wert durch 2 dividiert (Wortadressierung!) und auf 8 Bit verkürzt wird:
V = <*E* − . − 2>/ 2
Wenn *V* mit Vorzeichen größer als 8 Bit wird, so wird ein A-Fehler angezeigt.
Beispiele:
Folgende unbedingte Verzweigungsbefehle stehen auf der Adresse 120274(8):
a) in binärer Darstellung:
0 000 000 1   00 010 100
Operationskode   Verschiebung
in oktaler Darstellung: **0 0 0 4 2 4**
Dieser Befehl bewirkt zur Programmlaufzeit eine Verzweigung nach:
PC+2∗Verschiebung=
120276+2∗(+24)=120346
b) in binärer Darstellung:
0 000 000 1 11 111 100
in oktaler Darstellung: **0 0 0 7 7 4**
Dieser Befehl bewirkt zur Programmlaufzeit eine Verzweigung nach:
PC+2∗Verschiebung=**120276+2∗(−4)=120266**
c) in binärer Darstellung:
0 000 000 1 11 111 111
in oktaler Darstellung: **0 0 0 7 7 7**
Dieser Befehl bewirkt zur Programmlaufzeit eine Verzweigung nach:
PC+2∗Verschiebung=**120276+2∗(−1)=120274**
Sprung auf den Befehl selbst (endlose Schleife)!

Fortsetzung folgt!



1

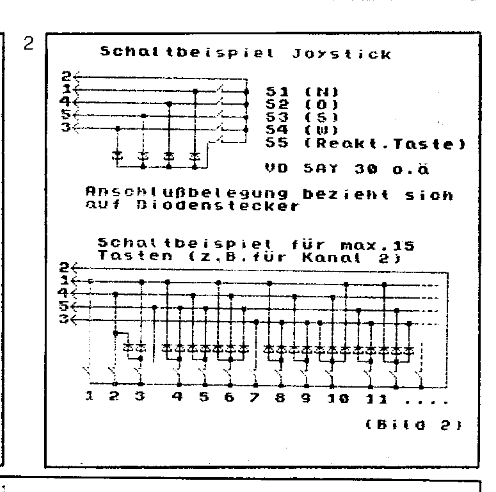
Joymodul

D1 DL021
D2 DL000
D3 DL010
D4/5 DL295
R9/10 100Ω
VD VQA13
VT SC236
R1-8 3,3K
R11 6,2K
R12 270Ω
C1 0,1µF

Belegung bezieht sich auf Dioden-buchsen

(Bild 1)

2



Joystickmodul für KC 85/2 (/3)

Hans-Georg-Werner, Berlin

Mit dem hier vorgeschlagenen Modul lassen sich die Einsatzmöglichkeiten des Kleincomputers erweitern. Dabei ist neben der Anwendung bei Reaktionsspielen auch die Verwendung bei Grafikprogrammen und die Eingabe externer Daten (2∗4 Bit) möglich.

Grundgedanke für die Realisierung war ein einfacher Aufbau und eine variable Anwendung. Untergebracht wurden Schaltung und Anschlußbuchsen in einem M005-USER-Modul. Die dazugehörige Beschreibung gibt auch Aufschluß über die Belegung des Steckverbinders. An der Frontblende befinden sich zwei Diodenbuchsen zum Anschluß der Spielhebel oder Tastatur (analog V24-Modul).

Die Schaltung besteht im wesentlichen aus zwei Funktionsgruppen: (vgl. Bild 1)

– Adreßdekoder zur Bildung des CS-Signals (Adr.,/RD,/IORQ)
– 2∗4 Bit Latch mit Tristate-Ausgängen (DL 295).

Im KC 85/2 (/3) stehen für den Anwender die I/O Adressen ab ØCØH zur Verfügung. Wegen der einfachen Dekodierung wurde für den Modul die Adresse ØFFH gewählt (D1). Zusätzlich wird /IORQ und /RD mit der Adresse verknüpft. Mit einem Gatter (DLØØ) und VT1 wird ein Monoflop realisiert, der eine Leuchtdiode an der Frontseite ansteuert. Diese nicht unbedingt notwendige Anzeige erlaubt aber Rückschlüsse bei fehlerhafter Programmabarbeitung, da jeder Aufruf des Moduls angezeigt wird. Um beim Einlesen durch eine IN-Operation die aktuellen Daten auf den Bus zu geben, muß die Übernahme in das Register vor der Aktivierung der Ausgänge erfolgen. Da die Adresse zeitlich vor den Steuersignalen anliegt, wird sie zur Steuerung des Übernahmesignals (T,Pin 9) herangezogen. Das eigentliche CS-Signal (OE,Pin 8) zur Aktivierung der Ausgänge wird durch die Verknüpfung der Adresse mit den Signalen /IORQ und /RD gebildet. Zur Schaltungsvereinfachung kann man auch Pin 9 mit Pin 8 verbinden, muß dann aber 2 IN Operationen ausführen, um die aktuellen Daten verarbeiten zu können.

Die Dateneingänge von D4 und D5 liegen über 3,3 kOhm auf L-Pegel. Sie werden durch die Tasten (Spielhebel oder Tastatur) über einen Schutzwiderstand (er verhindert bei eventuellen Kurzschlüssen eine Gefährdung des Computers) auf H gezogen. Um die gewünschten Funktionen (8 Richtungen, 1 Reaktionstaste) zu erreichen, muß bei der Realisierung des Spielhebels darauf geachtet werden, daß bei Zwischenstellungen jeweils 2 Kontakte geschlossen sind. Durch 4 Dioden, die über die Reaktionstaste an H gelegt werden, ergibt sich ØFH an den Eingängen. Diese bei der Richtungssteuerung nicht vorkommende Information wird als Reaktionssignal ausgewertet. Für zusätzliche Informationen lassen sich auch noch andere nicht vorkommende Codes erzeugen. Da für jeden Eingang 4 Bit zur Verfügung stehen, ist es auch denkbar, den 2. Kanal für Steuertasten zu nutzen. Durch einen einfachen Coder (siehe Bild 2) lassen sich bis zu 15 Tasten codieren. Auch eine Codierung aller 8 Bit wäre denkbar und für bestimmte Aufgaben sicher sinnvoll. Um das Modul an beliebiger Stelle im System unterbringen zu können, sind die Leitungen MEI/MEO(24B/24A) und IEI/IEO(10B/10A) durchzuschleifen.

Das hier vorliegende Programm wurde für die Nutzung von 2 Spielhebeln erstellt. In Verbindung mit BASIC wird es über die USER-Funktion aufgerufen. Dazu wird in die entsprechende BASIC-Arbeitszelle (772D) die Anfangsadresse eingetragen (512D; vgl. Bild 3).

Nach dem Aufruf des Maschinenprogrammes werden die Daten der Spielhebel eingelesen, nach linker und rechter Information getrennt und über eine Umcodierungstabelle geordnet. (1=N,2=NO,3=O,4=SO,.., 9=Reaktionstaste) Sie stehen auf 023EH (JOY 1, links) und 023DH (JOY 2, rechts) zur Verfügung. Anschließend erfolgt der Parameteraustausch von und an BASIC, je nach abgefragtem Hebel.

Prinzipiell läßt sich die Routine an beliebigen Stellen des freien RAM-Bereiches unterbringen, lediglich der IRM ist wegen der zusätzlich sichtbaren Zugriffe ungeeignet.

Für das gemeinsame Abspeichern beider Programme gibt es verschiedene Möglichkeiten. Am einfachsten ist das Einschreiben über POKE von BASIC aus. Durch Verschiebung des Programms (Abänderung der Merkzellen notwendig) läßt sich das Programm auch in 2 REM-Zeilen unterbringen, analog MP 7/87, S. 221.

Damit sind allerdings die Nutzungsmöglichkeiten dieser einfachen Schaltung bei weitem noch nicht ausgeschöpft. Zu nennen wäre z. B. die Datenfernerfassung u. ä. Dieser Beitrag sollte dazu als Anregung dienen.

3

```
END PASS 1

3000
3000                ;
3000                ;***************************************
3000                ;* Joysoftware mit Aufruf ueber USR-   *
3000                ;* Funktion. Anfangsadresse 512D in    *
3000                ;* Basic-Merkzelle 772D eintragen.     *
3000                ;*                              Werner *
3000                ;***************************************
3000                ;
3000                ;    *Abfrage-und Auswertungsprogr.*
3000                ;
3000                MERK    EQU 023CH       ;Merkzelle
3000                JOY1    EQU 023EH       ;Daten Hebel 1
3000                JOY2    EQU 023DH       ;Daten Hebel 2
3000                ABST    EQU 040H        ;Abstand Tabelle
3000                        ORG 0240H
0240 00                     DEFB 0H         ;Umcod.Tabelle
0241 01                     DEFB 1H
0242 03                     DEFB 3H
0243 02                     DEFB 2H
0244 05                     DEFB 5H
0245 00                     DEFB 0H
0246 04                     DEFB 4H
0247 00                     DEFB 0H
0248 07                     DEFB 7H
0249 08                     DEFB 8H
024A 00                     DEFB 0H
024B 00                     DEFB 0H
024C 06                     DEFB 6H
024D 00                     DEFB 0H
024E 00                     DEFB 0H
024F 09                     DEFB 9H
0250                        ORG 0200H
0200 2602                   LD H,02H
0202 DBFF                   IN A,(0FFH)     ;Abfrage
0204 DBFF                   IN A,(0FFH)
0206 323C02                 LD (MERK),A
0209 E60F                   AND 0FH         ;Maske Hebel 2
020B C640                   ADD ABST
020D 6F                     LD L,A
020E 7E                     LD A,(HL)
020F 323D02                 LD (JOY2),A
0212 3A3C02                 LD A,(MERK)
0215 0604                   LD B,04H
0217 CB3F           SCHIEB  SRL A           ;Div./16
0219 10FC                   DJNZ SCHIEB
021B C640                   ADD ABST
021D 6F                     LD L,A
021E 7E                     LD A,(HL)
021F 323E02                 LD (JOY1),A
0222                ;
0222                ;*Parameteraustausch von und an Basic *
0222                ;
0222 CD6FC9                 CALL 0C96FH     ;Par.ueberg.im
0225                                        ;DE-Reg.
0225 7B                     LD A,E          ;Par.auswertung
0226 FE01                   CP 1
0228 2809                   JR Z,JOYH1      ;Hebel 1 oder 2?
022A 3A3D02                 LD A,(JOY2)     ;Param.an Basic
022D 47                     LD B,A          ;NW-Teil Hebel 2
022E AF                     XOR A           ;HW-Teil Hebel 2
022F CDB1D0                 CALL 0D0B1H     ;FUWER an Basic-
0232                                        ;Variable
0232 C9                     RET
0233 3A3E02         JOYH1   LD A,(JOY1)     ;vgl.Hebel 2
0236 47                     LD B,A          ;NW-Teil Joy1
0237 AF                     XOR A           ;HW-TEIL Joy1
0238 CDB1D0                 CALL 0D0B1H
023B C9                     RET

ERRORS: 0000

DISPLAY 200 24F
   0200 26 02 DB FF DB FF 32 3C &_____2<
   0208 02 E6 0F C6 40 6F 7E 32 ____ào_2
   0210 3D 02 3A 3C 02 06 04 CB =_:<____
   0218 3F 10 FC C6 40 6F 7E 32 ?___ào_2
   0220 3E 02 CD 6F C9 7B FE 01 >__o_é__
   0228 28 09 3A 3D 02 47 AF CD (_:=_G__
   0230 B1 D0 C9 3A 3E 02 47 AF ___:>_G_
   0238 CD B1 D0 C9 FF 09 09 6D _______m
   0240 00 01 03 02 05 00 04 00 ________
   0248 07 08 00 00 06 00 00 09 ________

10 DOKE 772,512:!Eintrag Programmanfang
20 A=USR(1):B=USR(2):!Aufruf und Datenuebergabe an A und B
30 PRINT A,B
```

PC 1715-Tip

Textgestaltung durch Hoch- und Tiefstellung auf FX 1000/LX86

Der auf dem PC 1715 unter SCP lauffähige Textprozessor TPG13.COM ist ein sehr leistungsfähiges Dienstleistungsprogramm. Leider ist dieses trotz Installierung eines ähnlichen Druckers nur teilweise an die weit verbreiteten Drucker LX86 und FX1000 anpaßbar. Durch gezielte Veränderungen lassen sich jedoch die Probleme der Hoch- und Tiefstellung realisieren.

Folgende Arbeiten sind notwendig:

Installierung des Textprozessors mit einem ähnlichen Druckertyp, die Anwendersteuerzeichen ^Q ^W ^E ^R belegen mit Breit ein, Fett ein, Breit aus und Fett aus.

Dann ist die Datei TPG13.COM mit Hilfe von POWER zu verändern nach folgenden Schritten:

A>POWER
A=TYPEX TPG13.COM
(Auflisten des Hex Dump bis in der ersten Spalte die Adresse 0200 erscheint.)
A=READ spur sektor
(128-Byte-Block, der Adresse 0200 enthält, laden. Die Spur steht hinter T=.... , Sektor steht hinter S=....)
A=DS 8F
(Byte ab Adresse 8F werden verändert nach Tafel 1)

A=WRITE spur sekt.
(Abspeichern des Blockes)
A=TYPEX TPG13.COM
(Erneutes Auflisten des Hex Dump bis zur Adresse 06C0)
A=READ spur sektor
(Laden des Blockes, der die Adresse 06C0 enthält)
A=DS C1
(Byte ab C1 ändern nach Tafel 2)

A=WRITE spur sekt.
(Abspeichern des Blockes)
A=RUN TPG13

Das Beispiel gibt an, wie der entsprechende Text geschrieben werden muß:
Testbeispiel
Normal^VTief^V^RNormal
Normal^THoch^T^RNormal

Der hoch- oder tiefgestellte Text wird mit ^PV bzw. ^PT eingeschlossen und mit ^PR abgeschlossen.

Dagobert Mühlhaus

Tafel 1

Adresse	*8F*	*90*	*91*	*92*	*93*	*94*	*95*	*96*	*97*	*98*	*99*	*9A*	*9B*
Alt	*46*	*45*	*54*	*54*	*20*	*41*	*55*	*53*	*20*	*20*	*20*	*20*	*20*
ASCII	*F*	*E*	*T*	*T*		*A*	*U*	*S*					
Neu	*48*	*4F*	*43*	*48*	*2F*	*54*	*49*	*45*	*46*	*20*	*41*	*55*	*53*
ASCII	*H*	*O*	*C*	*H*	*/*	*T*	*I*	*E*	*F*		*A*	*U*	*S*

Tafel 2

Adresse	*C1*	*C2*	*C3*	*...*	*C6*	*C7*	*C8*	*...*	*DA*
Alt	*6A*	*0C*	*75*		*4A*	*0C*	*65*		*46*
ASCII	*j*	*^L*	*u*		*J*	*^L*	*e*		*F*
Neu	*53*	*30*	*00*		*53*	*31*	*00*		*54*
ASCII	*S*	*0*	*^§*		*S*	*1*	*^§*		*T*

KC 85/1-BASIC-Tip

Ändern von Zeichenketten

Das vorgestellte BASIC-Unterprogramm „ZK-KOR" ersetzt und erweitert den INPUT-Befehl bei der Eingabe von Zeichenketten. Dabei wird die zu ändernde Zeichenkette auf dem Bildschirm dargestellt, der Bereich bis zu einer maximal erlaubten Länge durch Unterstriche gekennzeichnet und der „Kursor" auf die erste Position der Kette positioniert.

```
500 REM "ZK-KOR"
505 POKE-4152,32:L5=LEN(EI¤):L1=0:POKE 37,0
510 L3=DEEK(45)-1:PRINT EI¤;STRING¤(L0-L5,CHR¤(227));:EI¤=""
520 IF L1>L5 THEN L5=L1
530 L1=L1+1
540 L2=4:L7=L1+L3:L4=PEEK(L7)
550 L2=L2+0.2:IF SIN(L2)<0 THEN POKE L7,207:ELSE POKE L7,L4
560 L6=PEEK(37):IF L6=0 THEN 550
570 POKE L7,L4:POKE 37,0:IF L6<>2 THEN 580
575 FOR L2=1 TO L5:POKE L3+L2,227:NEXT:L5=0:L1=1:GOTO 540
580 IF L6<>26 THEN 590
582 IF L5=L0 OR L1>L5 THEN BEEP:GOTO 540
584 FOR L2=L5 TO L1 STEP-1:POKE L3+L2+1,PEEK(L3+L2):NEXT
586 POKE L7,32:L5=L5+1:GOTO 540
590 IF L6<>31 THEN 600
592 IF L1<L5 THEN FOR L2=L1 TO L5-1:POKE L3+L2,PEEK(L3+L2+1):NEXT
594 IF L1<=L5 THEN POKE L3+L5,227:L5=L5-1:ELSE BEEP
596 GOTO 540
600 IF L6<>24 THEN 610
605 L1=L5+1:GOTO 540
610 IF L6<>25 THEN 620
615 L1=1:GOTO 540
620 IF L6<>9 THEN 630
625 IF L1<=L5 THEN 530 ELSE BEEP:GOTO 540
630 IF L6<>8 THEN 640
635 IF L1=1 THEN BEEP:GOTO 540:ELSE L1=L1-1:GOTO 540
640 IF L6<32 OR L6>127 THEN 650
642 IF L1<=L0 THEN POKE L7,L6:GOTO 520
644 BEEP:IF L1<L5 THEN 540 ELSE L4=32:GOTO 550
650 IF L6<>13 THEN BEEP:GOTO 540
652 IF L5>0 THEN FOR L2=1 TO L5:EI¤=EI¤+CHR¤(PEEK(L3+L2)):NEXT
654 IF L5<L0 THEN FOR L2=L5+1 TO L0:POKE L3+L2,32:NEXT
656 POKE -4152,16:PRINT:RETURN
```

Eine alternierende Darstellung von „Kursor" und dem von ihm verdeckten Zeichen gewährleistet dabei jederzeit die volle Lesbarkeit der Zeichenkette. Anschließend kann über die horizontalen Kursorsteuerzeichen auf das oder die zu ändernden Zeichen positioniert und die Zeichenkette geändert werden.

Das Unterprogramm unterstützt weiterhin die Funktionstasten DEL, INS und CL LN, sowie eine vorher zu definierende maximal zulässige Länge und selbstverständlich die ENTER-Taste. Nach Betätigen von ENTER wird die geänderte Zeichenkette vom Bildschirm gelesen und gespeichert.

Ein weiterer Vorteil des Unterprogrammes besteht darin, daß durch Variieren der Zeile 640 der Bereich der erlaubten Zeichen definiert werden kann. So kann beispielsweise durch

640 IF L6<>46 AND L6<48
  OR L6>57 THEN 650

der Bereich der erlaubten Zeichen auf die Ziffern 0 bis 9 und den Dezimalpunkt begrenzt werden. Fehleingaben werden akustisch signalisiert und übergangen.

Vor dem UP-Aufruf müssen definiert werden:

EI$ – die zu ändernde Zeichenkette
LO – die maximal erlaubte Länge von EI$.

Nach dem UP-Rücksprung beinhalten:

EI$ – die geänderte Zeichenkette.
L5 –die tatsächliche Länge von EI$.

Das Unterprogramm wurde auf der Schwarzweiß-Variante des KC 85/1 in der Monitorversion 01.02. erstellt und erprobt.

Gerd Kemnitz

Retten von Variablen

Gelegentlich kann es vorkommen, daß man eine Information über eine CLEAR-Anweisung oder einen Neustart hinweg bewahren möchte. Das kann man beim KC 85/3 dadurch erreichen, daß die zu rettende(n) Variable(n) (im vorliegenden Fall eine Zeichenkette) durch VPOKE-Anweisungen in einen Teil des Bildwiederholungsspeichers geschrieben wird/werden, der ja geschützt ist und daher durch CLEAR usw. nicht gelöscht wird. Im Beispiel war ein Name zu bewahren. Da der Speicherbereich von BC00H bis BFFFH durch eine Grafik-Zeichenliste belegt war, die Funktionstasten aber nur zum Teil benötigt wurden, kam der hintere Teil des Speichers für die Funktionstastenbelegung in Frage, und zwar im konkreten Fall ab Adresse B940H (siehe Programm-Auszug). Der erste Speicherplatz wird dabei mit der Länge der Zeichenkette belegt, die weiteren Speicherplätze mit den ASCII-Codes der einzelnen Zeichen. Der Aufruf der geretteten Zeichenkette erfolgt dann entsprechend durch VPEEK-Anweisungen. *Dr. Frank Steinmann*

Anmerkung der Redaktion: Das Verfahren funktioniert ebenfalls beim KC 85/2. Beim KC 87 muß anstelle der VPOKE- die POKE-Anweisung unter Beachtung des verfügbaren Speichers verwendet werden.

```
   .
   .
1000  INPUT„Name:";NN$
1010  E=LEN(NN$):VPOKE 14656,E
1020  FOR I=1 TO E
1030  F=ASC(MID$(NN$,I,1))
1040  VPOKE 14656+I,F:NEXT
   .
   .
2000  CLEAR
   .
   .
3000  PRINT AT(16,12);„Mach's gut,"
3010  E=VPEEK(14656)
3020  FOR I=1 TO E:F=VPEEK(14656+I)
3030  PRINT AT(17,11+I);CHR$(F):NEXT
3040  PRINT AT(17,12+E);„!..."
   .
   .
```

Laden von BASIC-Programmen aus ROM-Modulen

Dr. Hans-Jürgen Busch, Dresden

Für die robotron-Kleincomputer sind eine Reihe von Zusatzmoduln erhältlich, die die Anwendung auch für die Kleinrationalisierung erleichtern und das Einsatzspektrum wesentlich erweitern.

So besteht z. B. häufig der Wunsch, BASIC-Programme auf ROM-Moduln zu speichern, damit bei Routinearbeiten das mindestens tägliche Einlesen des gleichen Programms von Kassette entfallen kann, denn dieses wird bald als zu aufwendig und fehleranfällig empfunden.

Im folgenden werden Voraussetzungen, Mittel und Wege zum Erreichen dieses Zieles dargestellt.

Der Arbeitsspeicher des BASIC-Interpreters ist über Zeiger organisiert. Generierungsabhängig ist nur die niedrigste Adresse des Notizspeicherbereiches, diese liegt für die BASIC-Moduln und den KC 87 auf 300H.

Die Speicheraufteilung mit den entsprechenden Zeigern und ihre Verwendung zeigt Tafel 1.

Der Arbeitsspeicher des Interpreters einschließlich des Bereiches für das Quellprogramm liegt vollständig im RAM-Bereich des Rechners. Die Tabellen der einfachen und der Feldvariablen werden zur Laufzeit des Programms dynamisch hinter dem Quellprogramm aufgebaut, wobei der Abstand zum Stackpointer (SP) immer überwacht wird.

Beim Programmstart werden die genannten Tabellen gelöscht (nach RUN), d. h., die Zeiger DVARPT und

FSLPTR werden hinter das Ende des Quellprogramms gestellt (auf SVARPT). Damit wird deutlich, daß BASIC-Programme nicht in ROM-Bereichen lauffähig sind. Sie können aber mit Hilfe eines kurzen MC-Programms aus ROM-Bereichen in den Arbeitsspeicher des Interpreters geladen und dort abgearbeitet werden, wenn gleichzeitig mit dem Umladen die notwendigen Zeiger initialisiert werden.
Um zunächst ein BASIC-Programm auf einen ROM-Modul auszulagern, benötigen Sie:

- einen ROM-Erweiterungsmodul 690002.7, der das Programm dann aufnimmt
- eine entsprechende Anzahl von EPROMs U 2716 C
- einen Programmier-Modul 690023.6
- den Zusatzmonitor ZM von Programmkassette R 0112, R 0121 oder R 0122
- für große Programme einen RAM-Erweiterungsmodul 690003.5

Die Arbeitsschritte zum Auslagern sind:

1. Vorbereiten des Rechners (Programmier-Modul, RAM-Modul, BASIC, ZM)
2. Einlesen des auszulagernden BASIC-Programms von Kassette
3. Bestimmen der Länge des Programms
laenge = (SVARPT) – (PSTBEG).
Das Programm endet mit 0 0 0 und der Zeiger SVARPT zeigt hinter die dritte Null.
4. Bestimmen eines ausreichend großen freien RAM-Bereiches, der nach Möglichkeit mit einer glatten Speicheradresse (anfang) beginnt (unter Umständen auch im Freibereich des BASIC-Interpreters selbst).
5. Verschieben des BASIC-Programms ab (PSTBEG) in diesen Bereich auf Adresse anfang + 16H mit Hilfe des ZM
6. Eintragen des MC-Programms zum Umlagern von ROM in RAM mit Hilfe des ZM entsprechend Tafel 2.
7. Programmieren der EPROMs
8. Stecken der EPROMs auf die entsprechende Stelle des ROM-Moduls (Anfangsadresse start) – fertig!

Der ROM-Modul mit dem BASIC-Programm ist damit zur Nutzung vorbereitet. Wenn in den Rechner ein BASIC- und der ROM-Modul gesteckt werden, kann das MC-Programm zum Laden des BASIC-Programms aus der Kommando-Ebene mit
CALL start oder
CALL∗ start
aufgerufen und anschließend das BASIC-Programm mit RUN gestartet werden.

Tafel 1 ***Zeiger und ihre Verwendung***

Name	Adresse hex/dez.	Verwendung als Zeiger auf
PSTBEG	35F/863	Anfang des BASIC-Quellprogramms, RUN startet auf (PSTBEG)–1 (auf 0 initialisiert)
SVARPT	3D7/983	Anfang der Tabelle der einfachen numerischen und ZK-Variablen
DVARPT	3D9/985	Anfang der Tabelle der Feldvariablen (numerisch und ZK)
FSLPTR	3DB/987	Anfang des Freibereiches
STDPTR	356/854	Anfang Zeichenketten-Speicherbereich, (MEMSIZ)–1. Parameter von CLEAR, Initialisierungswert für SP
SWAPTR	3C4/964	Füllstand ZK-Speicherbereich (Füllung von hinten)
MEMSIZ	3B0/944	höchsten von BASIC verwendeten Speicherplatz, 2. Parameter von CLEAR, Ende ZK-Speicherbereich

Tafel 2 ***MC-Programm zum Laden von BASIC-Programmen***

```
21 .. ..  anfang: LD    HL,start+16H  ;Adr. nach RET dieses Programms
11 01 04          LD    DE,401H       ;Standardwert fuer (PSTBEG)
01 .. ..          LD    BC,laenge     ;entsprechend 3.
ED B0             LDIR                ;BASIC-Programm in
                                      ;Arbeitsspeicher laden
EB                EX    DE,HL         ;HL: 401H+laenge
22 D7 03          LD    (SVARPT),HL   ;Zeiger initialisieren
22 D9 03          LD    (DVARPT),HL
22 DB 03          LD    (FSLPTR),HL
C9                RET                 ;anschießend folgt das BASIC-Programm
```

KC 85/3-BASIC-Tip Maschinenprogramme (3)

Nach den beiden in /1/ vorgestellten Verfahren zur Einbindung von Maschinen- in BASIC-Programme noch ein drittes. Bei diesem Verfahren wird der Anfang des BASIC-Programmspeichers nach höheren Adressen verschoben, und der Bereich ab Adresse 401H steht zur Verfügung. Neben dem bereits erläuterten Zeiger für das Programmende (3D7/3D8H) muß hierbei der Zeiger für den Programmanfang (35F/360H) verändert werden. Man geht folgendermaßen vor:

1. Start des BASIC-Interpreters
2. Eingabe der Programmzeile:
10DOKE863,2560:RUN
2560 ist hier der neue Programmanfang. Damit steht der Bereich von 411H bis 9FDH zur Verfügung.
3. Einstellen des Programmendes mit:
DOKE983,2562
4. Ggf. Nullsetzen der Zellen 9FEH bis A00H mit POKE
(ist sofort nach dem Einschalten des Kleincomputers nicht erforderlich, da der Speicher gelöscht wird)
5. Starten des Programmes:
RUN
Danach steht der BASIC-Interpreter auf dem neuen Anfang und ist bereit zur Eingabe des Programmes. (überprüfbar mit LIST oder PRINT FRE(0))
6. Eingabe des Maschinenprogramms:
durch Eingeben mit MODIFY oder durch Einlesen eines mit dem Modul M027 Development erzeugten Maschinenprogramms
7. Eingabe des BASIC-Programms

Beim Abspeichern mit der CSAVE-Anweisung werden drei Teile gerettet:
– Programmzeile 10 (siehe oben)
– Maschinenprogramm
– BASIC-Programm

Nach dem Laden mit CLOAD und sofort folgendem Auflisten erscheint nur die eine Programmzeile. Mit RUN wird aber das Gesamtprogramm gestartet.
Die Vorteile sind:
– im Maschinenprogramm können 00-Bytes vorhanden sein
– die Adressen liegen fest
– das BASIC-Programm läßt sich beliebig bearbeiten (z. B. RENUMBER)
– Der verfügbare Bereich liegt nicht hinter dem Bildwiederholspeicher
– das Maschinenprogramm wird nicht gelistet.

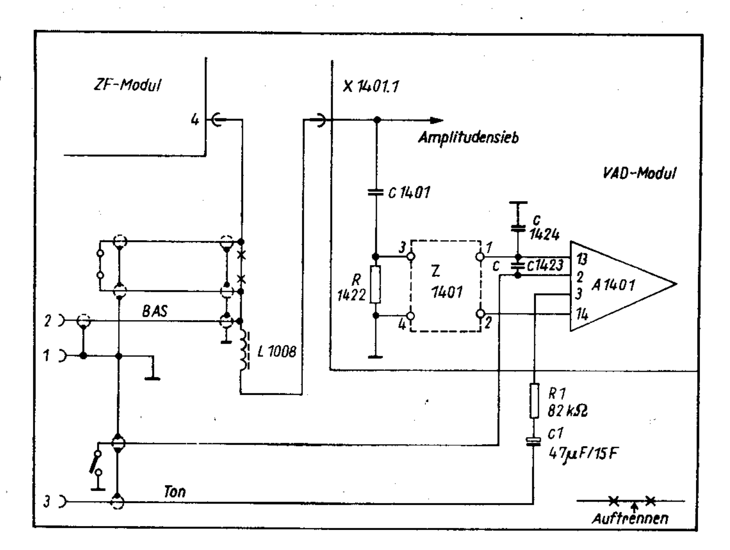
Klaus-Dieter Kirves

Literatur

/1/ Kirves, K.-D.: KC 85/3-BASIC-Tip Maschinenprogramme. Mikroprozessortechnik 1 (1987) 10, S. 308 u. 317

KC 85/3-Tip BAS-Ton-Anschluß am Robotron Combi-Vision RF 3301/ RF 3311

Als Ausgabegeräte für Kleincomputer kommen überwiegend handelsübliche Schwarzweiß-TV-Portable zum Einsatz.
Diese TV-Portable verfügen lediglich über einen HF-Eingang. Damit wird nur eine durchschnittliche Bildqualität erreicht. Eine Tonwiedergabe über TV ist nicht möglich.
Für viele Anwendungsfälle ist jedoch eine höhere Bildqualität erforderlich (/1/), und es wird die lautstärkegesteuerte Wiedergabe des Tonsignals gefordert.
Unter Nutzung des kombinierten FBAS-RGB-Ton-Ausgangs des KC 85/3(/2) und entsprechender Modifizierung des Combi-Vision kann man dieser Forderung gerecht werden.
In /2/ und /3/ wurde bereits die Modifizierung des Junost 402 B bzw. 401 B mit manueller bzw. automatischer Umschaltung des Bild- und Tonkanals behandelt.
Bild 1 zeigt einen Schaltungsauszug des Combi-Vision mit den notwendigen Anschlußpunkten für das BAS- und das Ton-Signal. Die Ankopplung des BAS-Signals erfolgt direkt an L 1008. Die des Ton-Signals erfolgt auf dem VAD-Modul.
Um eine lautstärkegesteuerte Ton-



KC-85-Tip Datenrecorder

Seit einiger Zeit ist der Datenrecorder LCR-C des VEB Elektronik Gera im Angebot. Dieser besitzt die Möglichkeit, den Antrieb des Recorders über den Motorschaltausgang des Kleincomputers direkt zu schalten. Das bringt einige Vorteile in der Bedienung. Von Nachteil ist allerdings, daß zum Spulen der Kassetten entweder eine VERIFY-Anweisung gegeben oder das Diodenkabel gezogen werden muß. Mit dem kleinen Programm, dessen HEX-DUMP in Tafel 1 dargestellt ist, ist es möglich, den Recorder ein- und auszuschalten. Das Menükommando lautet „LCR". Das Programm ist speicherverschieblich. Liegt es z. B. auf Adresse 0, so kann es vom BASIC mit CALL 6 aufgerufen werden.

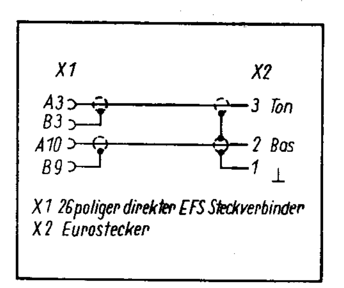
Klaus-Dieter Kirves

Tafel 1 HEX-DUMP des Programmes zum Schalten der Motorspannung

```
0000  7F 7F 4C 43 52 01 DB 88
0008  EE 40 D3 88 C9 00 00 00
```

wiedergabe zu realisieren, ist es erforderlich, den unbeschalteten NF-Eingang des A 223 zu nutzen.
Die Ankopplung erfolgt über C1 und R1 an Pin 3 des A 223. R1 dient zur entsprechenden Dämpfung des vom KC 85/3(/2) bereitgestellten NF-Signals. Pin 2 des A 223 wird zur Unterdrückung von Brummerscheinungen auf Masse gelegt.
Zur Umschaltung des Bild- und Tonkanals zwischen HF- und BAS-Einspeisung wurde beim aufgebauten Muster die am Combi-Vision vorhandene Kopfhörerbuchse verwendet. Möglich ist natürlich auch die Realisierung einer elektronischen Umschaltung nach /3/ unter Berücksichtigung der entsprechenden Anschlußpunkte im Combi-Vision.
Die Verbindung zwischen Kleincomputer und TV-Portable wird mittels 26-poligem direktem EFS Steckverbinder, 2adrigem abgeschirmtem Kabel und einem Eurostecker hergestellt. Bild 2 zeigt die Kontaktbelegung.

Detlef Bauer



Literatur

/1/ Schlenzig, St.; Schlenzig, K.: Ein Textverarbeitungssystem für KC 85/2. Radio, Ferns., Elektron. 35 (1986) 8, S. 501–502

/2/ Faulenbach, U.: FS Empfänger „Junost 402 B" – ein Monitor. Funkamateur 34 (1985) 12, S. 619

/3/ Kirves, K.-D.: Junost 401 B als Monitor für KC 85/3. Mikroprozessortechnik 1 (1987) 4, S. 126

/4/ Robotron Service-Anleitung. Combi-Vision RF 3301/RF 3311

Einlesen von Kassettendateien des ZX Spectrum auf KC 85/1

Der KC 85 hat ein dem ZX Spectrum ähnliches Kassettenaufzeichnungsverfahren. Deshalb ist es möglich, mit relativ geringem Softwareaufwand und ohne jegliche Hardwareveränderungen Dateien dieses Computers auf KC-85-Rechnern einzulesen.
Der logische Aufbau einer ZX Spectrum-Datei unterscheidet sich etwas von dem des KC 85/1. Eine Spectrum-Datei besteht i. A. aus einem Kopf, der Dateinamen, Typ, Ladeadresse und Dateilänge enthält, sowie einem einzigen Datenblock mit der im Kopf angegebenen Länge.
Kopf und Datenblock bestehen jeweils aus Vorton, Synchronbit, Daten und dem durch XOR-Verknüpfung gebildeten Kontrollbyte. Eine byteweise Synchronisation durch ein Trennzeichen wie beim KC 85 gibt es nicht. Entsprechend dem KC 85 werden die Zustände „0" und „1" jeweils durch eine Schwingung unterschiedlicher Tonfrequenzen codiert.
Das vorgestellte Programm stellt eine Minimalversion dar, die das Einlesen eines Blocks (Kopf oder Datenblock) realisiert und ins Betriebssystem zurückkehrt. Zur Initialisierung von PIOs und CTC werden Teile des Betriebssystems verwendet, teilweise durch direkten Aufruf, teilweise durch unveränderte Übernahme in das Ladeprogramm. Ebenfalls unverändert wird die Interruptroutine für das Lesen von Kassette benutzt. Die verwendeten Adressen gelten für die Betriebssystemversion 1.2 des KC 85/1, die auch im KC 87 eingesetzt ist.
Das Programm besteht aus der eigentlichen Leseroutine und einem Unterprogramm, das den durch die Interruptroutine veränderten Inhalt der Speicherzelle 6AH überwacht und an das Hauptprogramm übergibt. Im Falle eines CTC-Nulldurchgangs ohne Interrupt bricht das Unterprogramm den Lesevorgang ab und gibt die Steuerung ans Betriebssystem zurück.
Das Hauptprogramm verlangt zunächst 64 störungsfreie Halbperioden des Vortons. Anschließend wird die erste Halbperiode des Synchronbits erwartet. Bis dahin können noch beliebig viele weitere Vortöne auftreten. Nach der Erkennung des Synchronbits (wobei die zweite Halbperiode nicht ausgewertet wird) beginnt das Einlesen der Daten. Dabei fungiert das Register D als Bitzähler, während in E die Daten eingeschoben werden. Das Doppelregister BC kontrolliert die Dateilänge und bricht das Einlesen ab, wenn die auf den Adressen 36H, 37H vom Betriebssystem gespeicherte RAM-Obergrenze überschritten wird. Ist das Datenbyte in E vollständig, wird es in den RAM ab Adresse 4A0H ausgelagert. Die jeweils zweite Halbperiode jedes Datenbits wird überlesen, das Kontrollbyte nicht ausgewertet.
Da das Programm keine Weiterverarbeitung der gelesenen Daten vornimmt, setzt seine Anwendung das Vorhandensein eines Monitorprogramms zur Anzeige und Abspeicherung voraus. Dabei sollte beachtet werden, daß sich dieses Monitorprogramm durch Setzen der Adressen 36H, 37H vor dem Überschreiben mit Daten schützen muß.
Der zum KC 85/1 und KC 87 entwickelte Zusatzmonitor ist dafür gut geeignet.
Der Start des Einlesens erfolgt durch Aufruf des Namens RSPEC im Betriebssystem, während der Vorton der zu lesenden Spectrum-Datei bereits läuft. Nach Blockende erfolgt die Rückkehr ins Betriebssystem. Ein vorzeitiger Abbruch ist einfach durch Anhalten des Kassettenrecorders möglich.
Als Weiterentwicklung des Einleseverfahrens bietet sich die Umwandlung der vorliegenden Programmversion in ein Unterprogramm an, während ein neues Hauptprogramm zunächst den Kopf und anschließend mit der dem Kopf entnommenen Längeninformationen die Daten einliest. Möglich sind auch komfortable Programme, die unterschiedliche Dateitypen wie Textdateien, BASIC-Programme oder Assembler-Quelltexte einer Aufbereitung unterziehen, so daß sie mit den entsprechenden Systemprogrammen des KC 85 genutzt werden können. Solche Programme sind aber mit erheblichem Aufwand verbunden und erfordern trotzdem noch einige manuelle Nacharbeiten an den umzusetzenden Spectrum-Dateien.
Bei der Verwendung der Leseroutine auf KC 85/2 bzw. KC 85/3 ändern sich die Systemadressen, die gesamte Initialisierung und auch die Zeitkonstanten. Prinzipiell ist das vorgestellte Verfahren aber auf allen KC 85-kompatiblen Rechnern anwendbar.

Berthold Biener

Bild 1 Quelltext des Programms RSPEC

```
3000                    ORG  400H
0400 C32004             JP   LOAD
0403 52535045           DEFM 'RSPEC
040B 00                 DEFB 0
040C 3A6A00   HPLEN:    LD   A,(6AH) ;Halbperioden-
040F B7                 OR   A       ;laenge messen
0410 28FA               JR   Z,HPLEN ;kein Interrupt
0412 F5                 PUSH AF      ;Wert retten
0413 AF                 XOR  A
0414 326A00             LD   (6AH),A ;ARB loeschen
0417 F1                 POP  AF
0418 FEAB               CP   0ABH    ;sinnvolle Laenge
041A D8                 RET  C
041B F1                 POP  AF
041C C37D04             JP   LDEND   ;sonst Abbruch
041F 00                 NOP
0420 F3       LOAD:     DI           ;HAUPTPROGRAMM
0421 AF                 XOR  A
0422 326A00             LD   (6AH),A ;ARB loeschen
0425 CD0AFB             CALL 0FB0AH  ;Initialisierung
0428 D393               OUT  93H,A   ;PIO und CTC
042A D38A               OUT  8AH,A
042C 3E05               LD   A,5
042E D380               OUT  80H,A
0430 3EB0               LD   A,0B0H  ;Startwert CTC
0432 D380               OUT  80H,A
0434 3E0F               LD   A,0FH
0436 D38A               OUT  8AH,A
0438 3E0A               LD   A,0AH
043A D38A               OUT  8AH,A
043C 3EE7               LD   A,0E7H
043E D38A               OUT  08AH,A
0440 2A3600             LD   HL,(36H) ;RAM-Ende-Zeiger
0443 11A004             LD   DE,4A0H ;Pufferanfang
0446 B7                 OR   A
0447 ED52               SBC  HL,DE
0449 E5                 PUSH HL      ;Max.Pufferlaenge
044A EB                 EX   DE.HL
044B FB                 EI
044C 0640     SVTON:    LD   B,040H
044E CD0C04   VTON:     CALL HPLEN   ;Min.64 Vortoene
0451 FE70               CP   70H
0453 30F7               JR   NC,SVTON ;Kein Vorton
0455 FE30               CP   30H
0457 38F3               JR   C,SVTON ;Kein Vorton
0459 10F3               DJNZ VTON
045B CD0C04   SYNC:     CALL HPLEN   ;Sync-Bit suchen
045E FE70               CP   70H
0460 38F9               JR   C,SYNC  ;Noch Vorton
0462 CD0C04             CALL HPLEN   ;2.Halbperiode
0465 C1                 POP  BC      ;Pufferlaenge
0466 110008   BYTE:     LD   DE,0800H ;D=Bitzaehler
0469 CD0C04   BIT:      CALL HPLEN   ;E=Bytepuffer
046C FE7C               CP   7CH     ;Test 0/1
046E CB13               RL   E       ;Bit einschieben
0470 CD0C04             CALL HPLEN   ;2.Halbperiode
0473 15                 DEC  D
0474 20F3               JR   NZ,BIT  ;bis Byte komplett
0476 73                 LD   M,E     ;Byte in Puffer
0477 23                 INC HL
0478 0B                 DEC BC
0479 78                 LD A,B
047A B1                 OR C
047B 20E9               JR NZ,BYTE   ;Test Pufferende
047D CDE3FA   LDEND:    CALL 0FAE3H  ;Init Tastatur
0480 C34EF6             JP   0F64EH  ;Warmstart OS

ERRORS: 0000
```

RAM-Speichererweiterung für Z1013

Hans-Jochen Bachmann
Computer-Club robotron

Der Mikrorechner-Bausatz Z1013 ist aufgrund seiner übersichtlichen Schaltungsstruktur auch vom Amateur leicht zu einem hochwertigen Computersystem erweiterbar.
Die bisher gefertigten Varianten unterscheiden sich in folgenden Merkmalen:

- **Z 1013.01:**
  - *1 MHz Systemtakt*
  - *16 KByte dRAM*
  - *2 KByte ROM (Betriebssystem)*
  - *direkter Tastatursteckverbinder*
- **Z 1013.12:**
  - *2 MHz Systemtakt*
  - *1 KByte sRAM*
  - *2 KByte ROM (Betriebssystem)*
  - *direkter Tastatursteckverbinder*
- **Z 1013.16:**
  - *2 MHz Systemtakt*
  - *16 KByte dRAM*
  - *4 KByte ROM (Betriebssystem), von dem mittels einer Wickelverbindung vom Nutzer wahlweise nach Art des Tastaturanschlusses (Flachfolientastatur oder Alpha-Tastatur) jedoch nur 2 KByte ROM im Adreßbereich F000H bis F7FFH genutzt werden können.*
  - *indirekter Tastatursteckverbinder.*

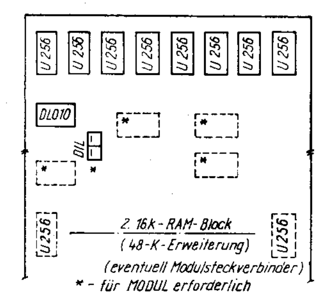


Bild 1 Bestückungsplan

Für alle Varianten sind die im folgenden vorgestellten Schaltungen als Anregung für Speichererweiterungen anwendbar.

1. RAM-Erweiterung bis 48 KByte

Der Aufbau wird zweckmäßig auf einer Lochraster-Leitplatte bzw. einer vom Amateurhandel vertriebenen trassierten RAM-Leiterplatte vorgenommen.
Die eigene Anfertigung einer Leiterplatte sollte nur von Versierten erfolgen, da die Trassierung nur auf einer begrenzten Fläche erfolgen kann.
Bewährt hat sich für den Amateur die manuelle Verdrahtung von Lochraster-Leiterplatten.
Abzuraten ist vom Aufsetzen der Zusatz-RAMs auf die bereits auf der Z1013-Platine vorhandenen Speicher-ICs; dies führt nach unseren Erfahrungen, bedingt durch Störspitzen in der Trassierung und der Spannungsversorgung, zu undefinierten Datenverfälschungen bei der Nutzung des gesamten verfügbaren Speicherbereiches.
Die Leiterplatte der RAM-Erweiterung kann entweder im „Huckepack" mit Plastabstandsbolzen auf der Originalplatine befestigt werden (Kompaktbauweise), damit entfällt der zusätzliche Einsatz von Multiplexern, oder sie wird als Erweiterungsmodul am System-BUS-Steckverbinder angeschlossen.

2. Speichererweiterung der Varianten Z1013.01 und Z1013.16

2.1. Erweiterung auf 32 KByte

Die Anordnung der Schaltkreise auf der Leiterplatte geht aus dem Bestückungsplan hervor (Bild 1). Die Verdrahtung erfolgt für alle RAM pingleich und ist aus dem Schaltplan (Bild 2) ersichtlich. Die Verdrahtung sollte sehr sorgfältig erfolgen und vor dem Anschluß an den Z1013 auf Schlüsse und Fehlverbindungenn geprüft werden, da die Fehlersuche bei Speicherfehlern sehr zeitaufwendig nach der Inbetriebnahme ist.

2.1.1. Anschluß bei Aufsatz auf der Z1013-Platine

Für den Anschluß bei dieser Variante hat sich als Verbindungsleitung das zum Lieferumfang gehörende Flachbandkabel und der Einsatz von Wickelstiften bewährt.
Die Wickelstifte werden an den Anschlußpunkten auf der Originalplatine eingelötet, dann wird mit Hilfe einer Wickelnadel der Anschluß des Kabels verwickelt. Es ist aber auch möglich, das Verbindungskabel direkt einzulöten.
Die Anschlußpunkte auf der Z1013-Platine sind in Tafel 1 aufgelistet. Der Vorteil dieser Variante besteht darin, daß die Adreßmultiplexer für die Speicherbereichauswahl auf der Z1013-Platine mitgenutzt werden können. Die geometrische Anordnung der Anschlußpunkte ist aus Bild 3 ersichtlich.

2.1.2. Anschluß als Modul am BUS-Steckverbinder

Der Aufbau erfolgt wie in Pkt. 2.1.1. beschrieben, jedoch sind zusätzlich folgende Bauelemente erforderlich:

- **2 DL 257**
- **1 DS 8286**
- **1 DL 010**
- **1 BUS**-Steckverbinder.

Die Schaltung zeigt Bild 4.

2.2. Erweiterung auf 48 KByte

Der Aufbau erfolgt analog Pkt. 2.1.; auf der Platine sind jedoch 16 × U256 statt bisher 8 × U256 in zwei Gruppen zu verdrahten. Alle Pins der zwei Gruppen, die nicht der Adreßauswahl dienen, werden miteinander pingleich verbunden. Die Adreßauswahl der jeweils zu 16 KByte zusammengefaßten Spei-

Tafel 1 Anschlußpunkte auf der Z1013-Platine

IC	PIN (Signal)			
A15	14(D0)	13(D1)	12(D2)	11(D3)
A16	14(D4)	13(D5)	12(D6)	11(D7)
A17	8(CASG)			
A24	8(/A15)	11(A14)		
A28	4(A0)	7(A1)	9(A2)	12(A3)
A41	4(A4)	7(A5)	9(A6)	12(WE)
A34	4(IMREQ)			

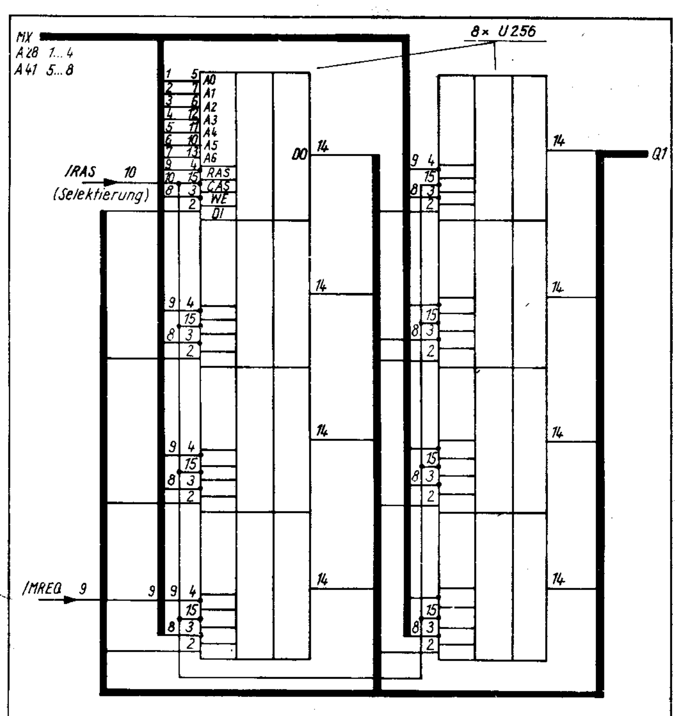


Bild 2 Schaltplan

Bild 3 Belegungsplan

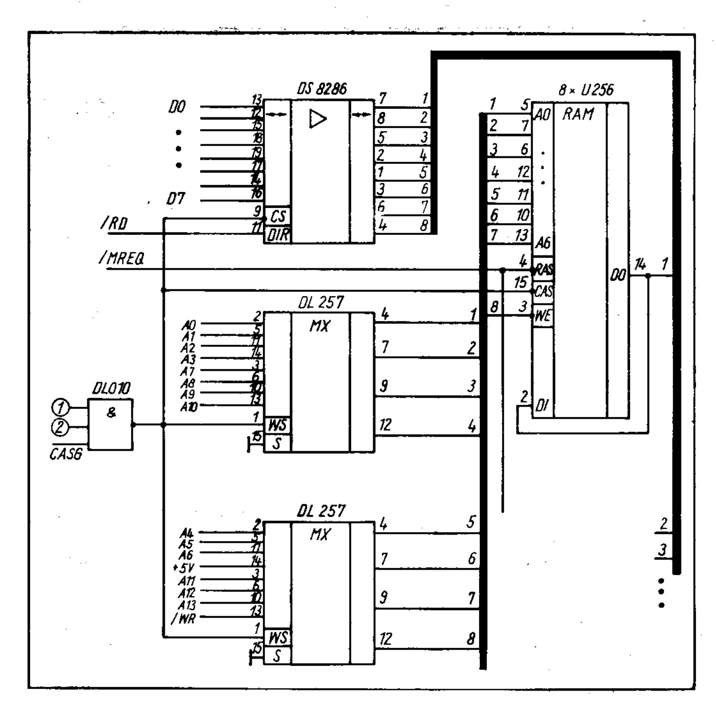


Bild 4 Anschluß des Moduls am BUS-Steckverbinder

Tafel 2 Speichererweiterung auf 64 KByte

Z1013.12	Z1013.01	Z1013.16
Industrievariante mit 1-KByte-RAM und 2 MHz Taktfrequenz	Amateurvariante mit 16-KByte-RAM und 1 MHz Taktfrequenz	Amateurvariante mit 16-KByte-RAM und 2 MHz Taktfrequenz
A28 u. A41 einlöten A10 – Lötbrücken entfernen u. bestücken Brücke E3 umlöten für 16-KByte-Aufrüstung 8 U256 einlöten für 64-KByte-Aufrüstung 8 U2164 einlöten und A15 u. A16 auslöten! weiter siehe Beschreibung	Taktbrücke E1 auf 2 MHz umlöten für 64-KByte-Umrüstung 8 U256 auslöten --- Schaltungsänderung lt. Beschreibung ---	für 64-KByte-Umrüstung 8 U256 auslöten 8 U2164 einlöten

Wird eine externe Speichererweiterung vorgenommen, ist eine höher belastbare externe Spannungsquelle einzusetzen und die interne Spannungsversorgung außer Betrieb zu setzen!

DL000
A14
A15
DIL

Bild 6 Speicherselektierung Modul

cherbereiche erfolgt wie nachfolgend beschrieben.

2.3. Adreß-Auswahl

Die Auswahl der Speicherbereiche erfolgt in 16-KByte-Blöcken.
Dazu werden die Signale
CASG, A14, A15
zur Selektierung für die Speicherbereiche
4000H ... 7FFFH
8000H ... BFFFH
(**C000H ... FFFFH** – nur bedingt!)
verwendet.

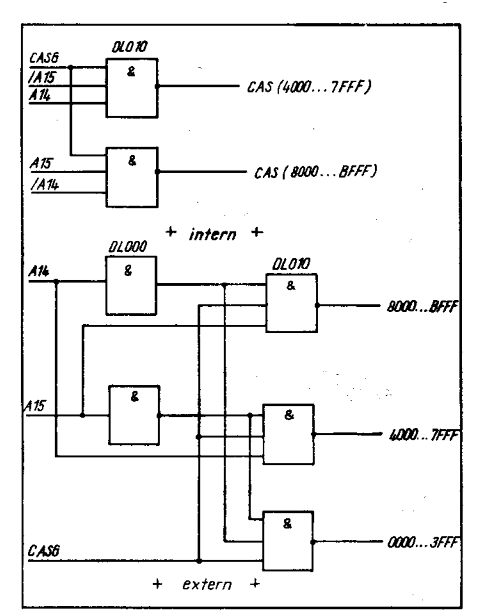
2.3.1. Aufsatz-Variante

Die Auswahl erfolgt über ein 3fach-NAND (z. B. DL010) und stellt das Signal CAS (low-aktiv) für die Selektierung der RAMs zur Verfügung (Bild 5).

2.3.2. Modul-Variante

Die Modulvariante für die Selektierung erlaubt ein Umschalten der Speicherbereiche mit DIL-Schaltern (Bild 6).
Die Adreßbereiche, die ausgewählt werden können, sind:

Bild 5 Speicherselektierung



4000H ... 7FFFH
8000H ... BFFFH
(**C000H ... FFFFH** – nicht nutzbar! – s. Text).
Der Vorteil besteht darin, daß jeder Modul wahlweise den möglichen Adreßbereichen zugeordnet werden kann.
Die Selektierung des Bereiches
C000H ... FFFFH
ist in der vorliegenden Variante nicht erlaubt, sie führt zu Doppelbelegung mit dem BWS und dem Betriebssystem und damit unweigerlich zum Systemabsturz.
Soll dieser Bereich mit Modul dennoch genutzt werden, so muß der Modul, der auf den Bereich C000H ... FFFFH adressiert ist, mit einer modifizierten Schaltungsänderung im Bereich EC00H ... EFFFH(BWS) und F000H ... F7FFH (Monitor) ausgeblendet werden (gilt generell!). Dazu wird das CS-Signal der ROM-Bereiche zusammengefaßt und zum Abschalten des RAM in diesem Adreßbereich genutzt (MEMDI).

3. Speichererweiterung Variante Z1013.12

Die Z1013.12 ist eine Industrievariante und gestattet problemlos die Aufrüstung auf 16 KByte oder 64 KByte RAM. Die erforderlichen Plätze für die RAM-ICs sind auf der Platine freigehalten.

3.1. Erweiterung auf 16 KByte

(s. Bild 7 in Verbindung mit den Schaltungsunterlagen)

4. Erweiterung auf 64 KByte

Die Erweiterung auf 64 KByte RAM kann kostengünstig mit 8 × U2164 erfolgen. Dazu ist es jedoch notwendig, eine geringfügige Trassierungsänderung auf der Originalplatine vorzunehmen.
Die 64-KByte-RAM-Erweiterung unterscheidet sich in den 3 Varianten des Z1013 nur geringfügig. Die Unterschiede sind in Tafel 2 zusammengefaßt.
Bei den Varianten, die bereits mit 16 KByte bestückt sind, sind mit einem Minischwalloder Entlötgerät die ICs vorher auszulöten. Die Schaltung der 64-KByte-RAM-Variante auf der Platine zeigt Bild 8.

Soll ein 64-KByte-RAM-Modul verwendet werden, ist auf der Originalplatine der gesamte RAM-Speicher wegzuschalten (MEMDI-Signal) und die vollständige Auswahlsteuerung auf dem Modul vorzusehen.
Die Umrüstung auf 64-KByte-RAM auf der Z1013-Platine bedingt die Unterbrechung von Leiterzügen und ein zusätzliches Verdrahten für die Selektierung. Diese Umrüstung sollte nur von sachkundigen Elektronik-Amateuren an Hand der Schaltungsunterlagen vorgenommen werden.

Bild 7 Erweiterung Industrievariante auf 16 KByte RAM

Bestücken:
1. D2, D7 – SY 360/2 (D2 wie D1, D7 entgegengesetzt)
2. C5.2, C5.3 – Elko 100 µF/16 V
3. V2 – Transistor SD 337
4. R35 – Widerstand 3,3K/0,125W
5. D5 – Diode SAY30 (SAY32/SAY40)
6. D6 – Z-Diode SZX 21/12
7. A28 u. A41 – IC DL257 (K531KP11/K555KP11)
8. A10 – IC DL074 (vorher Brücken auslöten)

Brücken um- bzw. einlöten:

9. E12

A33

10. E11

A38

11. E6 (+12 V)
12. Inbetriebnahme und Messen aller Spannungen an den RAM-Plätzen

00 5P
16 RAM 9
1 8
5N 12P

Sind alle Spannungen innerhalb der zugelassenen Toleranzen vorhanden, abschalten und weiter mit Pkt. 13.

13. Brücke E3 umlöten

A24

14. Bestücken A33 ... A40 – U256 (K565PY3, K565PY6)
15. Prüfen auf Schlüsse oder Haarrisse
16. Inbetriebnahme mit Monitor

Beim Einsatz von K565PY6 werden nur +5 V benötigt; es entfallen die Punkte 1 bis 6 und 11.

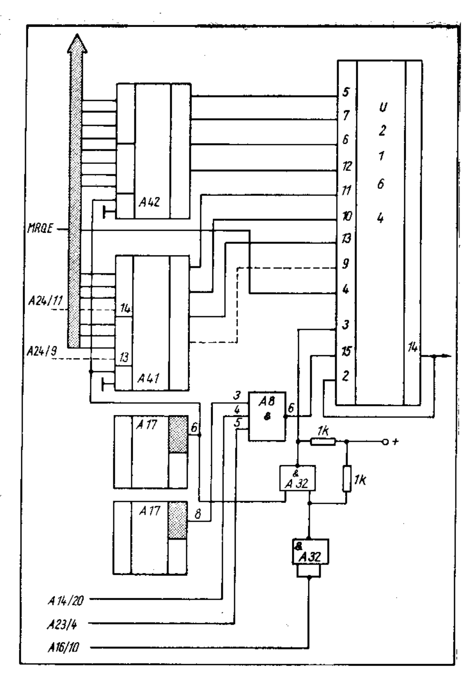


Bild 8 Z1013 mit 64 KByte dRAM

Arbeitsgänge:
- Auslöten der Brücken E6, E10, E12
- Auslöten von C 10.1 und C 10.2
- Versetzen der Brücke E11 um 2 Lötaugen in Richtung A39
- Alle Stützkondensatoren an den RAMs sind durch 100-nF-Keramikkondensatoren zu ersetzen.
- Der Stützelko für +5 V ist durch einen Elko 470 μF/6,3 V zu ersetzen.
- Auftrennen der Verbindung A24 (Pin 10) nach A8 (Pin 4)
- Auftrennen der Verbindung A24 (Pin 8) nach A8 (Pin 5)
- Trennen an A41 von Pin 13 (Pin 14) und mit A24 Pin 9 (Pin 11) verbinden
- von A41 (Pin 12) mit RAM (Pin 9) neu verdrahten
- an A8 (Pin 4 und 5) die /CS-Signale der ROMs legen
- Von A32 sind zwei Gatter noch frei, an die 2 Ziehwiderstände anzulöten sind und das /WR-Signal lt. Schaltung gelegt wird.

Die Umrüstung auf 64-KByte-RAM wurde auf ein Minimum an Änderungen auf der Rechnerplatine beschränkt.

5. Speichererweiterung mit 16-KByte-RAM-Moduln

Eine Speichererweiterung des Z1013 kann auch mit industriell gefertigten RAM-Moduln der Kleincomputer KC 85/1 oder KC 87 erfolgen, die vom Industrieladen des VEB Robotron-Vertrieb Erfurt vertrieben werden. Dazu ist eine BUS-Erweiterung für den Z1013 notwendig, die in Kürze als Baugruppenträger mit 4 Modul-Steckplätzen produziert und ebenfalls vertrieben wird.
Dies ist vor allem denjenigen Nutzern zu empfehlen, die nicht über ausreichende Elektronikkenntnisse verfügen. Zu beachten ist bei dieser Variante, daß auch die Stromversorgung erweitert werden muß, da das vorhandene Netzteil bereits die Leistungsgrenze erreicht hat (externe Spannungsbereitstellung).

6. Nachrüstung des Speicherbereiches E000H... E3FFH

Diese Nachrüstung wird allen denjenigen empfohlen, die keine 64-KByte-RAM-Erweiterungsvariante betreiben. Dazu sind keinerlei Veränderungen auf der Z1013-Platine notwendig; es sind nur die Plätze A15 und A16 mit statischen RAMs (U214/U224) zu bestücken.
Für die Industrievariante entfällt dies, dort sind diese beiden RAM-ICs bereits für den Speicherbereich 0H... 3FFH bestückt und können durch Umlöten der Brücke E3 auf den Bereich E000H... E3FFH verlegt werden (Bild 7).
Dieser Speicherbereich wird für die Nutzung von Dienstprogrammen, wie z. B. Druckerroutinen, Header Save/Load u. a., benötigt.

7. Schlußbemerkung

Der Beitrag zeigte Möglichkeiten und Anregungen für Speichererweiterungen auf, die den jeweiligen Bedingungen genügen.
Die Realisierung einer der vorgestellten Schaltungen bedingt grundlegende Kenntnisse und das Vorhandensein von Schaltungsunterlagen des MRB Z1013. Es wird darauf hingewiesen, daß Veränderungen der Originalschaltung in jedem Falle ein Erlöschen der Garantieansprüche zur Folge haben und auch u. U. zum Ausschluß von Reparaturleistungen (außer Pkt. 6) beim Hersteller führen können.
Die beschriebenen Veränderungen auf der Platine sollten deshalb nur von Elektronikern durchgeführt werden, die auch in der Lage sind, Reparaturen selbständig auszuführen. Im anderen Falle ist die Variante nach Punkt 5 vorzugsweise einzusetzen.

⊠ KONTAKT ☏

VEB Robotron-Anlagenbau, Computer-Club, PSF 180, Leipzig, 7010; Tel. 7 16 15 78

Hilfsroutinen zur Arbeit mit SCP-GX

Eberhard Schmidt
WTZ der Land- und Nahrungsgüterwirtschaft Frankfurt (Oder)

In /1/ wird ein leistungsfähiges TURBO-PASCAL-Programmpaket für die grafische Arbeit am A 7100 vorgestellt.
Der Service, den darauf aufbauende Anwendungslösungen bieten, läßt sich weiter erhöhen, wenn man die beiden Routinen gemäß Bild 1 einbaut.
Die logische Funktion „no_GDOS_GIOS" nutzt die Änderung des Interruptvektors auf Adresse 0380H durch GRAPHICS.CMD vom Eintrittspunkt des BDOS zum Eintrittspunkt des GDOS bei Initialisierung des Grafik-Modus.
Die Prozedur „KGS_Firmware_laden" ist die Umsetzung des Grafik-Vorbereitungskommandos L.CMD in TURBO-PASCAL-Befehle.

Literatur

/1/ Vetter, O.: Grafik am A 7100. Mikroprozessortechnik Berlin 1 (1987) 11, S. 325

⊠ KONTAKT ☏

WTZ der Land- und Nahrungsgüterwirtschaft, Abt. SP, Lebuser Chaussee 11, Frankfurt (Oder), 1200; Tel. 31 12 18


```
Listing of B:ARTIKEL.PAS                    DATE 17.12.87          PAGE 1

  1:  0 FUNCTION no_GDOS_GIOS : BOOLEAN;
  2:  0 (* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
  3:  0 (*  Liefert TRUE, wenn SCP-GX noch nicht mit dem Kommando  *)
  4:  0 (*                     GRAPHICS [d:]                       *)
  5:  0 (*  geladen wurde, sonst FALSE.                            *)
  6:  0 (* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
  7:  0 CONST  BDOS_Int =   224;
  8:  0        BDOS_Seg = $0104;
  9:  0        BDOS_Ofs = $0B06;
 10:  0 BEGIN
 11:  1    no_GDOS_GIOS :=
 12:  1          (MEMW[0:          BDOS_Int SHL 2  ] = BDOS_Ofs) AND
 13:  1          (MEMW[0:SUCC(SUCC(BDOS_Int SHL 2))] = BDOS_Seg)
 14:  1 END; (* no_GDOS_GIOS *)
 15:  0
 16:  0
 17:  0
 18:  0 PROCEDURE KGS_Firmware_laden(Version:BYTE;VAR Fehler:BOOLEAN);
 19:  0 (* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
 20:  0 (* Firmware der gewaehlten Versionsnummer vom akt.LW laden *)
 21:  0 (*   Wird die Firmware nicht gefunden, ist Fehler = TRUE   *)
 22:  0 (* z.B.:                                                   *)
 23:  0 (*   VAR nicht_gefunden : BOOLEAN;                         *)
 24:  0 (*   BEGIN KGS_Firmware_laden ( 6, nicht_gefunden ) END.   *)
 25:  0 (* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
 26:  0 CONST Status_ABS_KGS = $200;
 27:  0       Daten_ABS_KGS  = $202;
 28:  0       Recordlaenge   =  128;
 29:  0       allgemein      = 'GRAF';
 30:  0       Typ            =  'FRM';
 31:  0       Bit_1          = 2;
 32:  0 VAR   Firmware : FILE;
 33:  0       i        : BYTE;
 34:  0       Dma      : ARRAY[1..Recordlaenge] OF BYTE;
 35:  0 BEGIN
 36:  1    Fehler := TRUE;
 37:  1    ASSIGN(Firmware,allgemein+CHAR(Version+48)+'.'+Typ);
 38:  1    {$I-} RESET(Firmware); {$I+}
 39:  1    IF IORESULT > 0 THEN EXIT ELSE Fehler := FALSE;
 40:  1    REPEAT
 41:  2       BLOCKREAD(Firmware,Dma,1);
 42:  2       FOR i := 1 TO Recordlaenge DO
 43:  2          BEGIN
 44:  3           REPEAT UNTIL (PORT[Status_ABS_KGS] AND Bit_1) = 0;
 45:  3           PORT[Daten_ABS_KGS] := Dma[i]
 46:  3          END
 47:  2    UNTIL EOF(Firmware);
 48:  1    CLOSE(Firmware)
 49:  1 END; (* KGS_Firmware_laden *)
```

Bild 1 Listing

Schnelle Analog-/Digital-Wandlung und Sampling für 8/16-Bit-Computer

Ralph Drewello (23) erwarb 1982 das Abitur an der Spezialschule physikalisch-technischer Richtung in Jena. Danach arbeitete er als Vorpraktikant im Elektroniklabor des Institutes für Physiologie der Friedrich-Schiller-Universität Jena und absolviert seit 1983 ein Physikstudium an der Karl-Marx-Universität Leipzig.

Ralph Drewello, Jena

1. Vorbemerkungen

Im Ergebnis physikalischer Experimente erhält man häufig eine Vielzahl von Meßwerten, die in mehr oder weniger aufwendigen mathematischen Formalismen von einem Computer ausgewertet werden müssen. Die analogen Signale (Spannungen) müssen dazu mit hinreichender Genauigkeit digitalisiert und dem Computer für die Auswertung zur Verfügung gestellt werden. Dabei sind maximale Abtastrate und Auflösung der Analog-/Digital-(A/D)-Wandlung die entscheidenden Kriterien, die somit den Einsatzbereich einer solchen Schaltung bestimmen.
Die hier vorgestellte Lösung stellt ein flexibel einsetzbares Analoginterface für Mikrorechner dar, mit dem schnelle analoge Prozesse mit einer ansprechenden Genauigkeit aufgenommen werden können.
Seit April 1987 befindet sich im Institut für Neurobiologie und Hirnforschung der AdW der DDR, Magdeburg, eine realisierte Schaltung im Einsatz.

2. Zielstellung und Realisierungsmöglichkeiten

Als Ziel der Entwicklung wurde eine A/D-Wandlung mit einer Abtastrate von 1 MHz und einer Auflösung von mindestens 10 Bit gesetzt. Die digitalen Werte sollten von einem 8-Bit-Computersystem weiterverarbeitet werden können.
Aufgrund der gestellten Bedingung an die maximale Abtastrate konnte die direkte Übernahme der digitalisierten Werte mit dem 8-Bit-Computersystem nicht zur Anwendung kommen.
Einen 16-Bit-Prozessor einzusetzen, der praktisch als Einkartenrechner in Minimalkonfiguration sowohl die A/D-Wandlung und die Speicherung der Daten in einem eigenen Speicher als auch den Datentransfer zum Auswertesystem über eine parallele oder serielle Schnittstelle realisiert, wäre sicherlich eine sehr elegante Lösung des Problems. Auch ein mit 8 MHz getakteter 16-Bit-Prozessor ist aber nicht in der Lage, innerhalb einer Mikrosekunde einen 16-Bit-Wert von einem Port zu holen, ihn auf eine bestimmte Adresse im Speicher zu schreiben und die Adresse zu inkrementieren. Zum Beispiel braucht der 8086 allein für den IN-Befehl IN AX, DX (wobei DX die Portadresse enthält und in AX der 16-Bit-Wert eingelesen wird) schon 8 Takte = 1 μs bei einer Systemtaktzeit von 125 ns.
So verblieb nur noch die Möglichkeit, mit einem bestimmten Aufwand an diskreter Logik eine Schaltung zu entwerfen, die den geforderten hohen zeitlichen Bedingungen genügt.

3. Praktische Realisierung

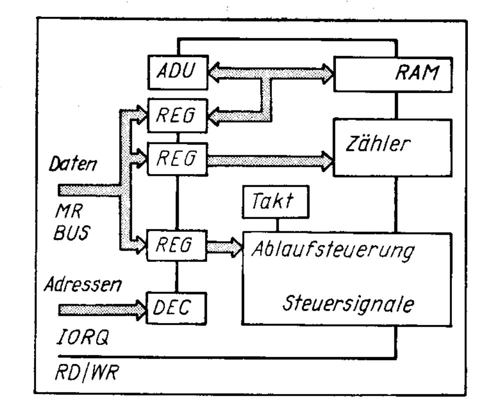
Das Prinzip der Schaltung besteht darin, daß die digitalisierten Werte vom A/D-Wandler kommend zunächst während einer Messung in einem eigenen Speicher aufbewahrt werden, wobei die Prozesse von einer speziellen Ablaufsteuerung (Hardware) realisiert werden.
Vor dem Start einer Messung wird die Schaltung zuerst mit den entsprechenden Steuerwörtern für Abtastrate, Anzahl der zu digitalisierenden Werte usw. vom Systemrechner aus programmiert. Mit dem Start der Messung durch einen OUT-Befehl übernimmt die Schaltung die Steuerung der A/D-Wandlung und der Speicherung der Daten und meldet sich nach Ablauf der Messung über ein gesetztes Bit an einem bestimmten Port zurück. Erst nach Abschluß einer oder mehrerer Messungen (Digitalisierung von jeweils n Werten) bzw. nachdem der Speicher gefüllt ist, werden die Daten vom Auswertesystem zur weiteren Verarbeitung übernommen.
Als Analog-/Digital-Wandler kam der sowjetische Typ K1108PW1 zum Einsatz[1]. Hier nun ein kurzer Überblick über die wichtigsten Daten des Bausteins:

- Wandlungszeit <1 μs für 10 Bit (<0,75 μs für 8 Bit)
- Wandlungsverfahren der sukzessiven Approximation
- interne Referenzspannung (äußere Referenz möglich)
- interner Taktgenerator (äußerer Takt möglich)
- internes Tristate-Register für die Datenausgänge
- Betriebsspannungen +5 V/–5,2 V (+–5%)
- Ausgangs- und Steuersignale sind TTL-kompatibel.

Bei der Konzeption der Schaltung wurde davon ausgegangen, nach Möglichkeit modernste Bauelemente einzusetzen, um die Anzahl der erforderlichen Schaltkreise möglichst gering zu halten. So wurden Speicherschaltkreise vom Typ UL6516 (2 KByte stat. RAM) verwendet. Die anderen Bauelemente stammen aus der aktuellen DL xxx- bzw. DS xxxx-Serie. Es wurden keine U880-spezifischen Interfacebausteine eingesetzt, so daß die Schaltung ohne Änderungen auch an einen 16-Bit-Computer angeschlossen werden kann. Eine 16-Bit-Version der Schaltung, die dann auch den 16-Bit-Datenbus in voller Breite ausnutzt, steht für die Zukunft zur Diskussion.
In Bild 1 ist das Blockschaltbild dargestellt. Der Quarzgenerator schwingt mit einer Frequenz von 10 MHz und liefert den Takt für die Ablaufsteuerung, mit der die diversen Steuersignale für den A/D-Wandler, die RAMs und die Zähler in einer bestimmten zeitlichen Reihenfolge generiert werden. Die Abtastrate des A/D-Wandlers ist im Bereich von 1,25 MHz bis 1,25 KHz programmierbar.
Sollten niedrigere Abtastraten erforderlich sein, besteht in der Betriebsart Einzelschritt die Möglichkeit, vom Systemrechner aus jeweils nur eine Wandlung zu starten und somit beliebig langsame Abtastraten zu realisieren.
Prinzipiell können auch andere A/D-Wandler zum Einsatz kommen, sofern sie die gleichen Anschlußbedingungen wie der K1108PW1 erfüllen. Dabei können bis zu 16 Bit breite Daten verarbeitet werden.
Die Schaltung wird direkt an den BUS eines 8- oder 16-Bit-Computers angeschlossen (es werden die Signale D0...D7, A0...A7, sowie RD, WR und IORQ verwendet) und belegt dort einen Block von 8 aufeinanderfolgenden Adressen im I/O-Bereich der CPU. Dieser Block ist dabei in 08H-Schritten in den Grenzen von 00H bis 0F8H frei verschiebbar. Der RAM-Speicher kann bis auf maximal 16 KWorte mit SRAMs UL6516–150ns bestückt werden. In diesem Fall könnten also 16384 digitalisierte Werte gespeichert werden.
Die Startadresse des Speicherbereiches, in den bei der folgenden Messung die digitalisierten Werte geschrieben werden, ist in Schritten von jeweils 128 Adressen frei programmierbar. Auch die Anzahl der A/D-Umsetzungen einer Messung ist in Schritten von je 128 Werten programmierbar. Dadurch ist es möglich, gezielt bestimmte Speicherbereiche zu beschreiben oder zu lesen bzw. auch mehrere Messungen hintereinander auszuführen. Die digitalisierten Werte werden über Eingabeoperationen vom Systemrechner aus gelesen. Dabei wird der Adressenzähler des externen Speichers automatisch inkrementiert, wodurch die Befehle INI und IND beim U880, die eine schnelle Datenübertragung ermöglichen, verwendet werden können.
Folgende Betriebsarten sind bei der vorgestellten Schaltung zu unterscheiden:

- Fortlaufende Digitalisierung von 128, 256, ...,16384 Werten mit Speicherung der Daten
- Einzelschritt-Digitalisierung mit Speichern der Daten
- Einzelschritt-Digitalisierung ohne Speicherung der Daten (sofortige Übernahme durch Systemrechner).

Bild 1 Blockschaltbild des A/D-Wandler-Interfaces



4. Software

Die Schaltung wird wie ein Interfacebaustein über eine Folge von Steuerworten initialisiert. Mit einem OUT-Befehl wird der Meßvorgang gestartet. Durch Polling wird das Ende der Messung festgestellt. Danach hat der Systemrechner Zugriff auf die Daten. Außerdem kann der RAM der Schaltung auch vom Systemrechner aus beschrieben werden und somit zur zeitweiligen Ablage von Daten dienen. Mit einem Minimum an Software, vorzugsweise in Assembler programmiert, kann die A/D-Wandlung relativ einfach in anderen Programmsystemen implementiert werden.

[1] Für die Überlassung des K1108PW1 bedanken wir uns an dieser Stelle herzlich bei unserem sowjetischen Partner aus dem Moskauer Energetischen Institut.

Wegbereiter der Informatik

MP hat sich entschlossen, mit dieser Ausgabe beginnend eine Reihe bedeutender Wissenschaftler in Kurzbiographien vorzustellen, die die Entwicklung der Computerwissenschaft durch ihre Arbeiten vorbereitet oder ganz unmittelbar bestimmt haben.
Dieses Vorhaben ist sehr zu begrüßen, denn es hat sich immer wieder gezeigt, daß auch heute, im Zeitalter des wissenschaftlichen Teamworks und der damit verbundenen wachsenden Spezialisierung des einzelnen, die leistungsfördernde Vorbildwirkung herausragender Persönlichkeiten keinesfalls aufgehoben ist. Im Gegenteil, gerade junge Wissenschaftler brauchen ihre Idole, durch deren Ausstrahlung sie in kürzester Zeit zu hocheffektiver Arbeit befähigt und immer aufs neue zu höchstem Einsatz motiviert werden.
Die biographische Serie von Dr. Klaus Biener ist dazu geeignet, einen kleinen Beitrag in dieser Richtung zu leisten. Sie zeichnet durch die Betrachtung einzelner Lebensverläufe ein erfaßbares Bild von dem inhaltlichen Reichtum der Computerwissenschaft sowie von der Dynamik und Schaffensbreite ihrer besten Vertreter und trägt in diesem Sinne auch zur Entwicklung des wissenschaftshistorischen Bewußtseins der Informatiker bei.

Prof. Dr. sc. techn. Jürgen Zaremba

BLAISE PASCAL

∗ 1623 Clermont, † 1662 Paris.



Im Mathematisch-Physikalischen Salon des Dresdner Zwingers kann man noch heute eine der ersten mechanischen Rechenmaschinen besichtigen, deren Erbauer der französische Mathematiker, Physiker und Philosoph B. Pascal gewesen ist. Pascal hatte im Laufe von 10 Jahren acht solcher Maschinen hergestellt, die reine Additionsmaschinen waren. Wie bei der damals vorhandenen Technik wohl kaum verwundern mag, ließ die Funktionssicherheit dieser Maschinen zwar einiges zu wünschen übrig, doch wurden ihre Konstruktionsprinzipien zum Vorbild auch für spätere Rechenmaschinen.
Schon in seiner Jugend fand Pascal in der familiären Umgebung ein förderliches wissenschaftliches Klima vor. Denn sein mathematisch gut vorgebildeter Vater, bei dem er auch den ersten Unterricht erhielt, pflegte eine enge, z. T. freundschaftliche Verbindung mit verschiedenen französischen Gelehrten, z. B. mit R. Descartes, P. de Fermat, M. Mersenne, G. P. de Roberval. Auch trafen sich im Pascalschen Hause Naturforscher wöchentlich zu wissenschaftlichen Gesprächen. Auf der Basis dieses Interessenkreises wurde übrigens – von Mersenne 1635 zunächst als „Freie Akademie" ins Leben gerufen – im Jahre 1666 die Pariser Akademie gegründet.
Von seinem 16. Lebensjahr an nahm auch B. Pascal an den Sitzungen dieser Freien Akademie mit teil. 1640 trat er bereits mit seiner ersten Veröffentlichung hervor, einer Abhandlung über Kegelschnitte. Darin beweist er eine von ihm gefundene Eigenschaft von Sechsecken, die einem Kegelschnitt einbeschrieben sind (*Pascalscher Satz, Pascalsche Gerade*).
Das sog. *Pascalsche Zahlendreieck* war schon früher bekannt, doch wurde es von Pascal bei Untersuchungen über die Koeffizienten des Binoms

$$(a + b)^n$$

für beliebige ganzzahlige n angewendet. Das Bildungsgesetz für diese Binomialkoeffizienten bewies er mit *vollständiger Induktion*, einem von ihm entwickelten, noch heute verwendeten mathematischen Beweisverfahren. Erstmals benutzte Pascal die Binomialkoeffizienten auch in der Kombinatorik. Seit 1654 korrespondierte er mit P. de Fermat über ein spieltheoretisches Problem; beide lösten es in verschiedener Weise und gelangten zu gleichen Ergebnissen; damit legten sie faktisch den Grundstein zur Wahrscheinlichkeitsrechnung.
Pascal hat sich auch mit physikalischen Fragestellungen befaßt. Der italienische Physiker E. Torricelli hatte 1644 sein berühmtes Experiment mit einer quecksilbergefüllten Röhre gemacht und damit fast gleichzeitig mit Otto v. Guericke in Magdeburg die Existenz der irdischen Lufthülle nachgewiesen. Um zu zeigen, *wie* die Höhe von Flüssigkeitssäulen von der Dichte der verwendeten Flüssigkeiten und vom Luftdruck abhängt, hat Pascal die Torricellischen Versuche mit Quecksilber, Wasser und Wein fortgeführt.
In neuerer Zeit wird ja ihm zu Ehren als Maßeinheit für den Druck die Bezeichnung 1 Pa verwendet. Übrigens ist auch die bekannte Programmiersprache PASCAL nach ihm benannt.
Im Jahre 1642 begann Pascal mit der Konstruktion seiner Rechenmaschine, das endgültige Modell wurde 1652 vollendet. Lange Zeit galt die Pascalsche Maschine als erste Rechenmaschine überhaupt. Neuerdings ist aber sicher erwiesen, daß der Tübinger Professor Wilhelm Schickard (1592–1635) im Geburtsjahr Pascals eine Rechenmaschine gebaut hat, die in Addition und Subtraktion eine einwandfreie Zehnerübertragung über sechs Stellen leistete und mit Hilfe einer verstellbaren Einmaleinstafel auch Multiplikation und Division ermöglichte. Ein Modell dieser Maschine befindet sich z. B. im Tübinger Rathaus. Näheres darüber wird in einem weiteren Beitrag zu berichten sein.

Dr. Klaus Biener

TERMINE

Fachtagung

Kommunikations- und Computertechnik

WER? Technische Universität Dresden, Sektion Informationstechnik, Wissenschaftsbereich Kommunikations- und Computertechnik
WANN? 14.–16. Februar 1989
WO? Technische Universität Dresden
WAS?
- Digitale Vermittlungstechnik
- Mikroprozessor- und Mikrocomputertechnik
- Lokale Kommunikationsnetze
- Funksignalübertragung und -verarbeitung
- Digitale Signalverarbeitung/Signalprozessoren
- Mikroakustische Signalverarbeitung
- Zuverlässigkeit von Kommunikations- und Computersystemen

WIE? Vortragsmeldungen bis 15. 5. 1988, Teilnahmemeldungen bis 15. 10. 1988 an:

Vorbereitungskomitee KomCom '89, Technische Universität Dresden, Sektion 09 Informationstechnik, Bereich Kommunikations- und Computertechnik, Mommsenstr. 13, Dresden, 8027; Tel. 4633941 oder 4633815

Dr. Finger

Fortsetzung von S. 108

des Fisches als G bestimmt. In Laborversuchen gelang die Bestimmung dieses Wertes mit einer relativen Standardabweichung von ca. 1%. Die Objekterkennung erfolgt durch Feststellen einer Schwellwertüberschreitung gegenüber den Referenzwerten der leeren Transportschale. Es wird ein Plausibilitätstest in 3 Bereichen mit jeweils 5 Bildpunkten vorgenommen. Erste Versuche unter einsatznahen Bedingungen ergaben für die Höhenbestimmung eine relative Standardabweichung von 2...4%. Problematisch ist die nicht ausreichend erfaßbare Korrelation zwischen maximaler Rumpfhöhe und Kopflänge sowie die Wirkung weiterer durch den Prozeß gegebener Einflußgrößen. Günstigere Werte werden durch die zusätzliche Vermessung und Einrechnung der totalen Fischlänge erwartet. Eine direkte Vermessung der Kopflänge ist bislang wegen des unscharfen Übergangs zum Rumpf nicht mit der notwendigen Sicherheit möglich.

Literatur

/1/ Schulz, K.-P.; Röhl, A.: Sichtsystem mit CCD-Zeilenkameras zur On-line-Vermessung und -sortierung bewegter Objekte. Diss. A., W.-Pieck-Universität Rostock, 1986
/2/ Schulz, K.-P.; Röhl, A.: Zweizeilenkamerasystem ZZK2. Technische Dokumentation zum A4-Bericht Merkmalerkennung. W.-Pieck-Universität Rostock, Sektion Technische Elektronik, 1985
/3/ Chung, T. L.: Entwicklung von Spezialprozessoren für die digitale Verarbeitung von Grautonbildern. Diss. A., Akademie der Wissenschaften der DDR, Fachbereich Mathematik/Kybernetik, Zentralinstitut für Kybernetik und Informationsprozesse, Berlin, 1984

KONTAKT

Wilhelm-Pieck-Universität Rostock, Sektion Technische Elektronik, Albert-Einstein-Straße 2, Rostock 1, 2500; Tel. 452 08

Technik international



Scanner ST 400 für Desktop-Publishing

Zu den neuesten Entwicklungen von Siemens zählt der Scanner ST 400. Er eignet sich besonders für Desktop-Publishing und für Archivierungsaufgaben.
Sämtliche Bedienungsfunktionen des ST 400 wurden in die Bedienoberfläche MS-Windows integriert. Damit kommt er vor allem für den Anschluß an AT-kompatible Personalcomputer in Frage. Über die international vereinbarte Schnittstelle SCSI (Small Computer Systems Interface) kann er aber prinzipiell auch an jeden anderen Rechner angeschlossen werden, der diese Schnittstelle bedient.
Der ST 400 ist ein Flachbett-Auflage-Scanner, der mit Faseroptik und LED-Beleuchtung arbeitet anstelle der sonst üblichen und störanfälligen Spiegel- und Linsensysteme. So konnten seine kompakten Abmessungen von 48 × 32 × 13 cm (L × B × H) erzielt werden. Der ST 400 bietet für die Graubildbearbeitung – etwa wenn bestimmte Bildteile stärker betont werden sollen – insgesamt 64 Quantisierungsstufen. Gleichzeitig ist auch die Bildauflösung zwischen 200, 300 und 400 dpi (15,85 Punkte/mm) frei wählbar. Der Scanner kann sowohl im Schwarzweiß- als auch im Halbton-Modus betrieben werden. Darüber hinaus hat der ST 400 viele Funktionen zum Definieren von Bildausschnitten. Bis zu 16 Fenster lassen sich mit verschiedenen Merkmalen versehen: Während des Scan-Vorganges kann z. B. ein Fenster als Graubild, ein anderes als Schwarzweißbild – ggf. mit einer anderen Auflösung – festgelegt werden.
Für eine A4-Seite braucht der ST 400 bei 400 dpi in schwarzweiß eine Scan-Zeit von weniger als 10 Sekunden. Es lassen sich sowohl die Formate A4, A5 und A6 als auch variable Formate scannen. Der eingebaute 2-Megabyte-Speicher ermöglicht eine vollständige Entkopplung von Scannen und Datenübertragung.
Für Desktop-Publishing-Anwender ist wichtig, daß die gescannten und aufbereiteten Daten direkt in ein entspechendes Layout-Programm eingebracht werden können.
Neben der für MS-Windows sind noch weitere Versionen der Scannersoftware geplant – so für die Bedienoberfläche GEM, für Unix (Sinix), VMS (DEC-Rechner) und Apples MacIntosh. (siehe auch 4. US).

Foto: Siemens

pa

Veränderungen des SCP1700

Das Betriebssystem **SCP1700** des **A7100** erlaubt bereits beim Systemladen die Einrichtung einer elektronischen Diskette (RAM-Disk) von über 300 KByte. Es ist auf einfache Art möglich, bei jedem Systemstart

- die RAM-Disk automatisch zu installieren und zu initialisieren
- den Kaltstart eines Programms auszuführen.

Die Veränderungen werden mit dem Programm POWER vorgenommen, dessen TPA bei **2080:4000H** beginnt. Auf diese Adresse wird die Systemdatei **SCP.SYS** geladen (Schreibschutz aufheben!):
LOAD SCP.SYS 4000
Ab Adresse 408AH wird das Kaltstartkommando in den CCP eingetragen:
ds 2080:408A
0CH 'SUBMIT START' 00H
Installation und Initialisierung der RAM-Disk wird durch Kurzschließen der entsprechenden Abfragen erreicht:

```
Adr.  Mnemonik                 Code
78EF  MOV BX,OFFSET ELDSK0     BB FD 35
wird zu
      JMPS +12H                EB 12
und
791B  MOV BX,OFFSET ELDSK1     BB 4B 36
wird zu
      JMPS +16H                EB 16
```

Das modifizierte System wird mittels **SAVE SCP.SYS 4000 125** auf die Diskette zurückgeschrieben.

Michael Lennartz

Molekular-elektronische Bauelemente – Schaltkreise des 21. Jahrhunderts?

Obwohl den gegenwärtigen Bauelementen auf Basis der bekannten Materialien im kommenden Jahrzehnt noch genügend Entwicklungsmöglichkeiten eingeräumt werden, mehren sich die Meldungen über Arbeiten an organisch-chemischen Substanzen, die für die Informationsverarbeitung geeignet erscheinen.
Als einer der maßgeblichen Verfechter der chemischen Schaltkreise wird der Chemiker Forrest L. Carter angesehen.
Eine Diode auf chemischer Basis könnte nach Carter wie folgt arbeiten: Wasserstoff besteht aus je einem positiv und negativ geladenen Teil. Da das Wasserstoffatom auf elektrische Felder reagiert, kann durch Potentialumkehr eine Diode realisiert werden. Als „Anschlußdrähte" für solche molekular-elektronischen Bausteine schlug Carter vor drei Jahren Schwefelnitrid-Kristalle vor. Von Schultz (Universität Stuttgart) werden sogenannte Radical Ion-Salze, z. B. Hexafluorarsenid, als Verbindungen zwischen molekular-elektronischen Bausteinen vorgeschlagen.
Bei der praktischen Realisierung einer Schaltung nach o. g. Prinzip treten jedoch u. a. folgende Probleme auf:
– Messung elektrischer Größen an Verbindungen mit Abmessungen von µm Länge und 10 nm Breite
– Temperaturabhängigkeit der Leitfähigkeit
– Sicherung der gewollten molekularen Verbindungen gegen zufällige Zerstörung.
Das Denkmodell von Carter, das der heutigen Realität weit vorauseilt, sieht vor, zunächst leicht aufbrechbare Großmoleküle in regelmäßigen (z. B. oktogonalen) Strukturen zu erzeugen. Zwischen solchen Großmolekülen bleiben ungenutzte Restflächen. Im nächsten Schritt bringt man die eigentlich gewünschten Strukturen auf, z. B. durch Aufdampfen. Dann werden die wie Masken wirkenden Großmoleküle durch chemische Prozesse wieder beseitigt und ein Array von nutzbaren Schaltelementen bleibt in den Zwickeln der entfernten Großmoleküle übrig. Diese Arrays von Schaltelementen kann man durch ein- oder zweidimensionale Kristalle untereinander verbinden.
G. M. Borsuk (Electronic Division Naval Research Lab, Washington) sieht bereits für 1994 die Realisierbarkeit solcher Bausteine in Nanometer-Strukturen. Zu diesem Zeitpunkt soll ein organischer Speicher mit einer Kapazität von 256 MBit und einer Zugriffszeit von 50 ps produzierbar sein.
Selbst wenn Abstriche von dieser kühnen Prognose gemacht werden und in einigen Jahren Speicher mit 256 MBit realisierbar wären, ist dies bereits als Sensation anzusehen. Auf einer Konferenz in Washington sollen von Vertretern unterschiedlicher Bereiche jedoch ähnlich lautende Prognosen gemacht worden sein.
Auch japanische Firmen sind auf dem Gebiet der Forschung für chemische Chips tätig. Eine Forschungsgruppe des japanischen Instituts für physikalische und chemische Forschung (RIKEN) berichtete über Erfahrungen mit Molecular Beam Epitaxie an organischen Substanzen. Drei Forschungsschwerpunkte wurden genannt: Der eigentliche Schaltungsaufbau (in der Reihenfolge der Hierarchie: Schaltelemente, Speicher und Computer), die Aufbau-Technologie und die erforderliche Nanometer Lithografie.
Das Institut soll außerdem an sogenannten Langmuir-Blodgett-Filmen arbeiten. Diese Vielschichtfilme aus Protein zeigen eine deutliche Abhängigkeit im U/I-Feld, wenn man sie dem Licht aussetzt. Der Dunkelstrom eines hergestellten 21-Schichten-Filmes folgt anderen Gesetzen als der Photostrom. Damit wären auch solche organischen Filme potentielle Schaltelemente, die mit Licht als „Trigger" arbeiten. Die Schaltzeiten liegen gegenwärtig im Sekundenbereich und sind damit nicht zu akzeptieren.
Die NTT Electrical Communications Laboratories berichteten von einem molekular-elektronischen Schalter, der im Labor bereits für einen Protonen-Transfer-Speicher verwendet wurde.
Hier baut man auf eine Nutzung des Tunnel-Effekts auf. Man verwendet Schichtfolgen von Al/Al_2O_3/organischen Substanzen/Pb, wobei die organischen Substanzen die Potential-Barriere der negativen Tunnelkennlinie hin- und herschieben können. Der Effekt ist reversibel und läßt sich durch Anlegen einer Plus- oder Minusspannung am Übergang nachweisen. Als organische Substanzen werden ultradünne Filme (z. B. aus 1-Nitroso-2 Naphtol oder 2-Nitroso-4-Sulfo-1 Naphtol) verwendet.
Von verschiedenen Forschungskollektiven werden Lösungen für das Einschreiben und Lesen der Informationen in die Moleküle vorgeschlagen. An der Case Western Reserve University verfolgt man eine elektrooptische Methode: Monolayer von etwa 100 000 Molekülen aus Phtalocyanin zeigen starke elektro-optische Effekte. Der intramolekulare Ladungstransfer beim Ein- und Ausschreiben soll $< 10^{-12}$ s betragen, Schaltzeiten von 10^{-13} s sollen beobachtet worden sein.
Reine molekular-elektronische Bauelemente (MED) werden voraussichtlich erst im nächsten Jahrhundert angeboten werden. Die überraschenden Fortschritte der letzten Jahre bei den Grundlagen dürfen die Langfristigkeit der Arbeiten nicht hinwegtäuschen. Die größten noch zu lösenden Probleme sind die Kombination von Prozeßtechnik, Geschwindigkeit, Adressierbarkeit sowie die Einbindung in die bestehende Umwelt.

Wi

Quelle: VDI-Nachrichten Magazin 9/87



Börse

Liebe Leser,
ein mittlerweile sehr großes Angebot an Lösungen zeigt uns, daß wir mit dem kostenlosen Service der Börse „ins Schwarze" getroffen haben. Wir freuen uns, daß diese Rubrik bei Ihnen von Anfang an so großen Anklang gefunden hat, und zahlreiche Hinweise belegen auch, daß – nicht zuletzt aufgrund der Auflagenhöhe der MP – die gewünschten Effekte zu verzeichnen sind. Da wir uns besonders bei dieser Rubrik um eine hohe Aktualität bemühen, möchten wir Sie an dieser Stelle bitten, künftig folgendes zu beachten:
Abgesehen davon, daß es nicht zulässig ist, identische Manuskripte gleichzeitig mehreren Zeitschriften anzubieten, führt diese – wenn auch gutgemeinte – Praxis einiger Betriebe dazu, daß sich insgesamt der Zeitraum bis zur Veröffentlichung vergrößert. Da es bezüglich der Aktualität also sinnvoller ist, viele Lösungen einmal zu veröffentlichen als wenige Lösungen mehrmals, sollten Sie sich entscheiden, in welcher Fachzeitschrift Ihr Angebot am effektivsten plaziert ist. Als „Postwurfsendungen" an verschiedene Redaktionen erkennbare Lösungen werden wir also im Interesse einer Nachnutzung aller Angebote nicht mehr zusätzlich veröffentlichen. Wir glauben, damit die höchsten volkswirtschaftlichen Effekte erzielen zu können und bitten – in diesem Sinne – um Ihr Angebot.

Ihre Redaktion MP

Erweiterter Druckeranschluß für PC 1715

Vielfach besteht der Wunsch, Drucktechnik aus dem Bestand vorhandener Rechner (Bürocomputer, K1630 usw.) auch am PC 1715 einzusetzen. Im allgemeinen verfügen diese Drukker über einen IFSS-Anschluß. Um die kosten- und zeitaufwendige Nachrüstung des PC 1715 mit der IFSS-Interface-Steckeinheit zu umgehen, wurde ein **Adapter V.24 – IFSS** entwickelt, der auf den V.24-Steckverbinder X5 des Grundgerätes aufgesteckt wird. Der Adapter hat die Größe 50 × 50 mm^2 und realisiert die für das DC1/DC3-Protokoll erforderliche Sende- und Empfangsstromschleife. Aus diesem Grunde ist der abgerüstete PRINTER-Ausgang X4 des PC 1715 für die Anpassung an IFSS-Geräte nicht geeignet. Ein entsprechender Treiber steht im SCP zur Verfügung und muß lediglich installiert werden. Damit wird auch der Betrieb von zwei gleichzeitig an das Grundgerät angeschlossenen logische Drukkern über verschiedene logische Gerätebezeichnungen möglich. Mit Hilfe des Adapters können Typenraddrukker 1152, Seriendrucker 1157 oder auch der Plotter 832 (ZWG) problemlos angesteuert werden.
Zur Herstellung repräsentativer Schriftstücke bietet sich der Anschluß einer elektronischen Schreibmaschine als Schönschriftdrucker an. Für das preisgünstige Modell S6009 wurde eine **Interfacesteckeinheit** zum Anschluß an den PRINTER-Ausgang des PC 1715 entwickelt. Die Steckeinheit wird direkt auf den Teststeckverbinder der S6009 gesteckt und innerhalb des Gehäuses untergebracht. Durch Wickelbrücken ist entweder V.24 (Standard) oder IFSS einstellbar. Neben den erforderlichen Interfacebausteinen enthält die Steckeinheit einen 2-KByte-EPROM (Steuersoftware) und die U856 (SIO). Der Druck erfolgt aus einem Puffer mit einer Geschwindigkeit von ca. 12 Zeichen/s. Der Befehlsumfang der seriellen Schnittstelle ist zu dem des Centronics-Interfaces für die S6009 kompatibel. Beim Einbau der Steckeinheit wird die S6009 auf den Stand 6050 umgerüstet (Erweiterung des Funktionsumfangs im Betrieb ohne Interface). Beide Lösungen stehen zur Nachnutzung als Funktionseinheiten zur Verfügung. Der Adapter V.24 – IFSS kann in geringen Stückzahlen kurzfristig bereitgestellt werden.

Zentralinstitut für Kernforschung Rossendorf, Abt. KFM, PSF 19, Dresden, 8051

Dr. Fromm

Software für Rechnerkommunikation

Zur Realisierung des Datentransfers zwischen verschiedenen Rechnern wurden folgende Softwarepakete entwickelt:
DTUU Kopplungssoftwarepaket zum Dateitransfer zwischen einem UDOS-Rechner und einem UNIX-Rechner. Die UDOS-Seite ist in PLZ, die UNIX-Seite in der Hochsprache C realisiert. Damit ist die Anpaßbarkeit an alle UDOS- bzw. UNIX-Rechner mit einer V.24- bzw. IFSS-Schnittstelle gewährleistet. Neben dem Datentransfer ist die Anwendung des UDOS-Rechners als Terminal des UNIX-Rechners möglich.
DTSU Kopplungssoftwarepaket zum Dateitransfer zwischen einem SCP(CP/M)-Rechner und einem UNIX-Rechner. Die SCP(CP/M)-Seite dieses Softwarepaketes ist in TURBO-PASCAL, die UNIX-Seite in C entwickelt worden und besitzt zu DTUU analoge Eigenschaften. Die Anpaßbarkeit an verschiedene SCP(CP/M)-Rechner (z. B. BC A 5120, PC 1715, A 7100 o. ä.) ist auch bei diesem Paket gewährleistet.
DTSS Filetransferprogramm für zwei 8-Bit-SCP(CP/M)-Rechner über die V.24- bzw. IFSS-Schnittstellen. Der Filetransfer erfolgt mit 9600 Bd (V.24) bzw. mit 4800 Bd (IFSS). Dieses Softwarepaket ist sehr effektiv beim Datentransfer zwischen Rechnern mit Diskettenlaufwerken verschiedener Formate und Größen.

VEB RFT Nachrichtenelektronik Leipzig „Albert Norden", Abt. EK5, PSF 15, Leipzig, 7027; Tel. 6 83 32 20

Bruchertseifer/Buchwald

Selbstdefinierte Zeichen bequem generieren

Das kleine BASIC-Programm ZEIGEN (ZEIchenGENerator) unterstützt effektiv die Selbstdefinition von Zeichen am KC 85/2 (/3). Dazu wird in einem Bildschirmfenster in 64facher Vergrößerung ein Punktraster vorgegeben (8 × 8), in dem alle 1-Bits mit dem Zeichen X und führende 0-Bits mit einem beliebigen Zeichen (außer Komma) belegt werden müssen.
Nach Abschluß des Bitmusters werden die für ein Zeichen erforderlichen 8 Bytes im Speicher abgelegt, und auf dem Bildschirm erscheint die Codezahl sowie der Ausdruck des Zeichens in Originalgröße. Falls dieses Zeichen nicht den Vorstellungen entspricht, kann es wieder gelöscht werden. Das ist schon alles! Die notwendige Speicherorganisation übernimmt das Programm. In der Grundeinstellung werden die neuen Zeichen ab Code 61H = 97 (Kleinbuchstaben) generiert.
Die neuen Zeichen stehen sofort nach ihrer Generierung zur Verfügung. Beschränkt man sich auf die Änderung der 26 Kleinbuchstaben, dann sind alle anderen Zeichen des Zeichensatzes gleichzeitig vorhanden. Selbstverständlich können auch andere Sequenzen des Zeichensatzes geändert werden, dazu ist lediglich eine Adresse in der ersten Programmzeile zu aktualisieren.
Das Programm ist 2 KByte lang, es beinhaltet außer den genannten Funktionen das Maschinenprogramm COPY, so daß man die Zeichentabellen im Speicher verschieben kann.
Interessenten stellen wir ZEIGEN kostenlos zur Verfügung; wir bitten aber um Zusendung eines adressierten und frankierten Rückumschlags und einer leeren Kassette.

EOS Arnstadt, Schloßplatz 1, Arnstadt, 5210

Bamberger

Abrechnung, Analyse und Auswertung der Fuhrparkleistungen

Das Programm wurde für Büro-, Personal- und Arbeitsplatzcomputer mit der Programmiersprache TURBO-PASCAL entwickelt.
Neben einer universellen Stammdatenpflege von Kunden-, Fahrer- und Fuhrparkdatei sowie der Bewertung nach PAO 370 Teil A, Stundenverrechnungssätze nach AO 151 und betrieblichen Werten ist eine Funktionsauswahl nach
– Bewertung von Leistungsnachweisen entsprechend Frachtbrief
– Rechnungsdruck über Rechnungsvordruck 910955 oder formlos
– Monatsabschluß für Buchungsanweisungen und innerbetriebliche Fahrzeug- und fahrzeugbezogene Auswertung
– Zahlungsaustrag
möglich.
Für den multivalenten Einsatz ist eine universelle Laufwerkspezifikation für alle Daten vorhanden.

ZBE Landbau Meißen, Korbitzer Straße, Meißen-Korbitz, 8250

Krömer/Dr. Schiller

Dateitransfer zwischen dBASE II und dBASE III

Um eine Datei-Kompatibilität zwischen den weit verbreiteten Datenbanksystemen dBASE II (REDABAS) und dBASE III (REDABAS M16) herzustellen, wurde ein komfortables Programm entwickelt, welches die Konvertierung von *.DBF-Dateien zwischen beiden Systemen in beide Richtungen ermöglicht. Einschränkungen hinsichtlich der Dateigrößen bei dBASE II werden berücksichtigt.
Das Programm DBT.EXE liegt in kompilierter Form vor und ist unter MS-DOS (ab 3.0) bzw. DCP 3.1 lauffähig. Es sortiert *.DBF-Dateien im aktuellen Verzeichnis automatisch nach Zugehörigkeit zum jeweiligen Datenbanksystem und stellt die Files in getrennten Fenstern dar.
Folgende Befehle werden ausgeführt:
CURSOR-Tasten
Blättern im Verzeichnisfenster, Fensterwahl
COPY (C)
Kopieren von *.DBF-Dateien zwischen Verzeichnissen und Laufwerken. Wenn nötig, Ergänzung im Verzeichnis-Fenster.
RENAME (R)
Umbenennen von *.DBF-Dateien mit Änderung der Eintragung im Verzeichnis-Fenster.
INFORM (I)
Abfrage aller verfügbaren Informationen über eine *.DBF-Datei und Darstellung im entsprechenden Fenster.
DELETE (D)
Löschen von *.DBF-Dateien mit Korrektur der Eintragungen im Verzeichnisfenster.
TRANSFER (T)
Dateikonvertierung mit automatischer Richtungserkennung und Veränderung der Einträge in den Verzeichnis-Fenstern.
Bei Überschreitung der Parameter einer dBASE II-Datei wird keine Konvertierung vorgenommen.
Für die Konvertierung einer ca. 20 KByte großen Datei werden ca. 8 Sekunden benötigt.

VEB Kombinat Wälzlager und Normteile, BT Forschung/Entwicklung Leipzig, Merseburger Straße 8, Leipzig-Rückmarsdorf, 7101; Tel. 47 43 91

Dr. K. Löschke

Wir suchen ...

... zur Nachnutzung eine Hard- und Softwarelösung zur Rechnerkopplung K 1510 – PC 1715/AC 7100 oder Anschluß eines Magnetbandkassettengerätes K 5261 an einen PC 1715/AC 7100.

VEB Bekleidungswerke herdas Greiz, Friedhofstr. 1, Greiz, 6600; Tel. 27 41

Bratsch



Entwicklungen und Tendenzen

Super-Chip

TRW Inc. (USA) entwickelt im Rahmen des staatlichen amerikanischen VHSIC-Projekts einen „Super-Chip", der 35 × 35 Millimeter mißt und 18,8 Mio. aktive Elemente aufnehmen soll. Die Arbeiten sind so weit fortgeschritten, daß jetzt 4 der geplanten 29 Makrozellen der Gesamtschaltung produktionsreif sind. Diese Makrozellen des komplexen DSP (Digital Signal Processor) können u. a. auf einen SRAM mit einer Kapazität von 2 MBit zurückgreifen. Das ist ein Wert, der doppelt so hoch liegt wie bei den einzeln hergestellten 1-MBit-RAM. Bei der Herstellung wird eine CMOS-Technik benutzt.
Im Rahmen der „Phase 2" des VHSIC-Programms sind bereits die ersten Produktionsmaschinen, die im Submikrometerbereich arbeiten, an anderer Stelle in Betrieb genommen worden (Motorola u. a.). Der Schritt zu 0,5-Mikrometer-Geometrie gilt als Grenze, die aus technischen Gründen bis auf weiteres nicht unterschritten werden kann. 0,25 Mikrometer gelten als physikalisch möglicher Grenzwert. Ein weiteres Grundproblem der „Super-Chips" oder der „wafergroßen IC", die hohe Ausschußquote, wurde von TRW mit neuen Mitteln gelöst. Dazu gehört eine „automatische Software-Konfiguration", die abhängig vom Zustand des Chips arbeitet; dazu kommt eine eingebaute Testfunktion. Damit können unter Ausnutzung eingebauter Redundanz Defektstellen umgangen werden.

pa
aus Neue Zürcher Zeitung

Superintelligente Mikroship-Karte wird getestet

Zuerst gab es die Kreditkarte, dann kam die „intelligente Karte". Jetzt wird die nächste Generation von intelligenten Kreditkarten – eine „superintelligente Karte" – getestet. In dieser Karte sind ein Mikrocomputerchip und ein rechnerähnliches Tastenfeld kombiniert, mit ihr können Kredit-, Scheck- und Sparkonten belastet und andere Funktionen ausgeführt werden. Die Karten sind mit einer LCD-Anzeige ausgestattet und werden mit einer Lithium-Batterie betrieben.
Wie das Unternehmen *Visa International* bekanntgab, wird die Firma, die die „superintelligente Karte" gemeinsam mit dem Elektronikkonzern Toshiba entwickelt hat, sie bei 2000 japanischen Kartennutzern testen. Im Gegensatz zu den einfachen „intelligenten" Karten soll die „superintelligente" Ausführung völlig unabhängig eingesetzt werden können.

pa

32-Bit-PC von NEC und Fujitsu

Die NEC Corp. stellte kürzlich ihren ersten 32-Bit-Personalcomputer vor. Der neue Personalcomputer PC 9812 arbeitet mit dem 32-Bit-80386-Mikroprozessor von Intel mit einem 16-MHz-Taktgenerator und wird durch den 16-Bit-V30-Mikroprozessor von NEC ergänzt, um die Kompatibilität mit den restlichen Personalcomputern der PC-9801-Reihe zu sichern.
Der PC-9812 verfügt über einen Hauptspeicher mit 1,5 MByte, der auf 14,5 MByte erweitert werden kann. Er ist mit zwei Laufwerken für 5 1/4-Zoll-Floppy-Disks mit einer Kapazität von einem MByte und einem 40-MByte-Hard-Disk-Laufwerk ausgestattet. Der Neue hat zwei grafische Betriebsarten: Eine hochauflösende mit 1120 × 750 Punkten und eine „normale" mit 640 × 400 Punkten, so daß der Computer die für andere Personalcomputer der PC-9801-Reihe erarbeitete Software verwenden kann. NEC bietet verschiedene Betriebssysteme für den PC-9812 an, darunter MS-DOS (Version 3.1) in japanischer Sprache, UNIX-PC-UX/V (Version 3.0) und MS-OS/2 in japanischer Sprache.
Fujitsu Ltd. stellte einen 32-Bit-Personalcomputer vor, der ebenfalls mit einem 80386-µP von Intel arbeitet. Der neue PC heißt FMR-70KD und hat einen 2-MByte-RAM, der bis auf 10 MByte erweitert werden kann, ein 5,25-Zoll-1-MByte-Floppy-Disk-Laufwerk und ein 3,5-Zoll-40-MByte-Hard-Disk-Laufwerk.

pa

Transportabler Mikrorechner mit Braille-Schrift

Hexabraille ist ein Mikrorechner für Blinde sowie auch für den normalen Gebrauch, der in Frankreich vom Laboratorium des Conservatoire National des Arts et Metiers entworfen und hergestellt wurde. *Hexabraille* ist transportabel und kann netzunabhängig mindestens 14 Stunden betrieben werden. Es setzt sich aus dem Mikrorechner *Epson HX-20* und einem Braille-Anzeige-Modul zusammen. Die Software Coppelius (Version 3.0) ermöglicht die Auswahl zwischen der Azerty- und der Braille-Tastatur, die Programmierung in BASIC oder in Assembler, das Erfassen und Bearbeiten von Texten, das Drucken auf herkömmlichen oder Braille-Druckern usw.

pa
aus PC-Informatique

ICs statt Disks

Die amerikanische Firma Mesa Electronics Inc. bietet Steckkarten an, die mit Halbleiterspeichern bestückt sind und mit einer Kapazität von derzeit 1024 KByte kleinere Plattenspeicher ersetzen sollen. Die Schaltung ist so ausgelegt, daß sie für den PC-Bus von IBM kompatibel ist und mit CPUs einer Taktfrequenz von 8 MHz betrieben werden kann. Auf der Karte enthaltene Firmware kann als Ersatzsteuerung für Plattenspeicher angesehen werden. Sie gestattet Kompatibilität mit Standardapplikationsprogrammen. Je nach Auslegung der Schaltung werden RAM, EPROM und EEPROM verwendet. Es wird durchgehend CMOS-Technik genutzt, und der Leistungsbedarf liegt im aktiven Zustand mit weniger als 100 Milliwatt entsprechend niedrig. Der Hersteller sieht Anwendungsmöglichkeiten in rauhen Umgebungsbedingungen, in denen die üblichen Disk-Laufwerke über Gebühr belastet würden (Staub, Stoß, Temperatur). Die Speicherkarte soll sich bei Temperaturen zwischen −40 und +85 °C problemlos betreiben lassen. Ein weiterer Vorteil ist der geringe Raumbedarf.

pa
aus Neue Zürcher Zeitung

Denkende und handelnde Chips

Wissenschaftlern der AT & T Bell Laboratory (USA) soll es gelungen sein, Computer-Chips zu entwerfen, die das Wiederauffinden gespeicherter Informationen und das Lösen von Problemen ähnlich den Gehirnzellen nachahmen. Die Forscher hoffen, mit diesen Chips spezialisierte Maschinen bauen zu können, die Aufgaben „erkennen" und „sich erinnern".
Die elektronischen Nervennetzwerke (ENN) bestehen aus kompakten, hochintegrierten Komponenten, wie Widerständen und Verstärkern. Ähnlich sollen die künstlichen „Nerven" arbeiten, kontinuierlich und gemeinsam, um Antworten schnell zu erhalten.
Der komplexeste Chip mit gegenwärtig 256 elektronischen „Nerven" ist 1/4 Zoll groß, hat über 25 000 Transistoren und mehr als 100 000 Widerstände. Frühere Silikon-Chips hatten nur wenige Dutzend „Nerven". In der Entwicklung befinden sich noch andere Chips, wie ein 54er elektronischer „Nerven"-Chip mit durch Software programmierbaren Widerständen und eine 484-Widerstandsmatrix, die zum Testen neuer Mikroherstellungsprozesse genutzt wird, und die eine größere Schaltungsdichte in kleineren Bauelementen realisiert.
Zur Zeit befindet sich die erste Generation der 54er Chips im Test. Sie soll genutzt werden, um Bit-Muster, wie Namen, zu speichern, wiederaufzufinden und sie innerhalb von Sekundenbruchteilen abzurufen. Die Grundelemente der Schaltungen bilden Widerstände. Eine Widerstandsmatrix verbindet Verstärker (Neuronen). Bei einer elektrischen Abfrage antworten die Chips wie ein Bündel von Nerven. Forscher von AT & T erwarten, daß ENN-Chips Informationen in ungefähr 400 ns wiederauffinden. Damit wären sie schneller als biologische Nerven.

wi
aus Designs News 3/87

IBMs neues VM

IBM hat das Virtual Machine/Extended Architecture System Product (VM/XA SP) angekündigt, ein Betriebssystemn der oberen Leistungsklasse, das IBMs gegenwärtiges VM/XA ersetzen wird. IBM weist darauf hin, daß das neue VM die Architektur der Systeme der oberen Leistungsklasse, wie die des 3090 Modells 600 E, voll ausnutzt. Das neue VM unterstützt das Conversational Monitor System (CMS), und IBM versichert, daß es volle VM-Kompatibilität auf den System/370-Prozessoren mit erweiterter Architektur von IBM gewährleistet. Neue VM/CMS-Funktionen schließen die Unterstützung gleichzeitiger Operationen von System/370 und erweiterter Adressierung ein. IBM erklärt, daß VM/XA auch bis zu 4mal mehr Zentralspeicher und doppelt soviel Kanäle unterstützt, als das vorangegangene VM-System.
Wenn VM/XA SP genutzt wird mit der neuen Einrichtung der 3090 E-Serie, der Multiple High Performance Guests Support, können bis zu 4 Gastbetriebssysteme gleichzeitig auf Prozessoren der E-Serie laufen. Neue Interfaces gestatten die Entwicklung von VM-Anwendungsprogrammen, die portabel sind zwischen System/370 und System/370-Einrichtungen mit erweiterter Architektur, die unter VM/XA SP laufen. Die ursprüngliche Unterstützung von IBMs SNA wird gewährleistet.

Wi
aus Datamation 16/87

Zugriff in 25 Nanosekunden

Der japanische Hersteller Toshiba Corp. will im Herbst dieses Jahres mit einem 1-MBit-SRAM mit einer Zugriffszeit von 25 Nanosekunden auf den Markt kommen. Toshiba sieht vorzugsweise Anwendungen bei transportablen Endgeräten, die nur wenig elektrische Leistung aufnehmen dürfen. Mit der 0,8 Mikrometer-Mikrolithographie können etwa 6,3 Mio. Elemente, wie Transistoren und Widerstände, auf einer Chipfläche von 6,86 × 15,37 mm^2 integriert werden, was dem Integrationsgrad von 4 MBit eines dynamischen Schreib-/Lesespeichers entspräche. Die Zellengröße wird mit 5,6 × 9,5 µm angegeben.

Wi
aus Blick durch die Wirtschaft 6. 5. 87

Intel gab endgültige Spezifikationen für den Mikroprozessor 80486 bekannt

Intel hat die endgültigen Spezifikationen für den 80486 bekanntgegeben, der als Mikroprozessor der nächsten Generation betrachtet wird. Er soll spätestens 1990 fertig sein und wird in low-power CMOS realisiert werden. Er wird einen 32-Bit-Datenbus und das Äquivalent von 1 bis 1,25 Mio. Transistoren enthalten. Im Gegensatz dazu haben der 80386 250 000 und der 80286 135 000 Transistoren. Ein Rechner, der den 80486 einsetzt, wird die Leistungsfähigkeit eines IBM-Rechners der Sierra-Serie (IBM-3090-Serie) besitzen und 20 Mio. Befehle pro Sekunde ausführen können. Die Kompatibilität mit den Vorgängern 8086, 80186, 80286 und 80386 soll gewährleistet sein.
Der Mikroprozessor wird verbesserte Möglichkeiten der Mensch–Maschine–Kommunikation besitzen, die Bildschirme hoher Auflösung mit sofort verfügbaren Fenstern erfordern, gesteuert mittels Hardware und künstlicher Intelligenz.

Wi
aus Dataquest 7/87

Einchipmikrorechner

Von H. Kieser und M. Bankel. VEB Verlag Technik, Berlin, 1. Auflage, 320 Seiten

Das vorliegende Buch stellt eine umfassende Einführung in die Technik der Einchipmikrorechnerfamilie U881, U882, U883, U884 und U886 dar.
Auf den ersten einhundert Seiten werden die interne Struktur, das Zeitverhalten, der Befehlssatz, die Steuerregister und das Interruptverhalten der Einchipmikrorechner (EMR) ausführlich erläutert. Dem Leser, bei dem in diesem Teil des Buches bereits Grundkenntnisse der Mikrorechentechnik vorausgesetzt werden müssen, werden alle wesentlichen Kenntnisse über das flexible Ein-/Ausgabesystem, die Vielzahl der programmierbaren Peripheriefunktionen sowie über die Register- und Adreßraumstruktur vermittelt.
In einem weiteren Abschnitt werden Hinweise zur Programmierung der EMR gegeben. Zugriffe auf den Registersatz sowie auf den externen Speicher werden in verschiedenen Varianten und anhand von Beispielen erläutert. Besonderen Wert wurde auf die Programmierung der Zähler-/Zeitgeberfunktionen und der seriellen Schnittstelle gelegt. Ein Programmbeispiel zur automatischen Bitratenerkennung rundet dieses Kapitel ab.
Sehr ausführlich und anhand vieler Beispiele wird die Problematik der arithmetischen Verarbeitung behandelt, die sicher für einen großen Anwenderkreis der EMR bedeutsam ist. Dem Programmierer wird eine Programmbibliothek für häufig auftretende Arithmetikaufgaben zur Verfügung gestellt.
Für den Einsatz der EMR in komplexeren Mikrorechnersystemen werden Ergänzungsbauelemente und Komponenten zur Systemerweiterung vorgestellt. Der Hardwareentwickler erhält hier alle notwendigen Informationen zum Anschluß externer Speicher, bipolarer Standardbauelemente, von Systemelementen anderer Mikroprozessorsysteme (U880) sowie von Schnittstellen für analoge Signale. Die übersichtlichen Schaltungsbeispiele können als Standardlösungen für viele Anwendungen übernommen werden.
Ein Abschnitt beschäftigt sich ausführlich mit den weiteren Typen der EMR-Familie. Der EMR U883, dessen interner Speicher mit einem BASIC-Interpreter und einer Bootstrap-Routine zur Initialisierung der internen Peripheriefunktionen maskenprogrammiert ist, stellt für die Anwender, für die sich aus Effektivitätsgründen die Herstellung einer eigenen Maske nicht lohnt, eine günstige Alternative dar. Seine Eigenschaften und Anwendungsgebiete, der TINY-MP-BASIC-Interpreter sowie die hardwareseitige Entwicklungsunterstützung werden besonders ausführlich dargestellt.
Ein Applikationsbeispiel zum Einsatz des U882 in einem UKW-Empfänger mit PLL-Frequenzsynthese rundet das gelungene Buch ab. Hervorzuheben ist auch die Begriffserklärung am Schluß des Buches, die nicht nur für die Anwender der EMR-Familie informativ ist. Der Anhang enthält eine Befehlsliste, eine Tabelle der Operationskodes, Pinbelegungen (auch der angeführten Peripheriebauelemente), Übersichten zu den Steuerregistern sowie zur Syntax des U883-MPBASIC.
Das Buch wendet sich an Praktiker, an industrielle Anwender der EMR, aber auch an Hobbyelektroniker und Studierende. Sowohl der Hardwareentwickler als auch der Programmierer findet eine geschlossene Darstellung der Technik der EMR vor. Die vielen Beispiele veranschaulichen und bereichern die Darlegungen und sind oft sofort in eigene Applikationen übertragbar. Einen besonderen Vorzug des Buches stellen die vielen kleinen, sachdienlichen Hinweise und Programmiertechniken dar, die sonst in keiner anderen Literaturstelle auffindbar sind. Derartig geschlossene, sprachlich ausgereifte, mit praktischen Beispielen, Tafeln und Übersichten bereicherten Zusammenstellungen sollte es für jede Mikroprozessorgeneration in unserem Buchhandel geben.

Ulrich Wiesner

UNIX für Führungskräfte – ein umfassender Überblick

Von Y. Skirota und T. L. Kunin. Springer Verlag Berlin–Heidelberg–New York–London–Paris–Tokyo 1987, Reihe: Springer Compass, 210 S., ISBN 3-540-16905-9

UNIX wurde unter Leitung von Ken Thompson und Devis Ritchie 1969 in den Bell Telephone Laboratories entwickelt.
Dieses leistungsfähige Betriebssystem, das multi-user- und multi-tasking-fähig ist, zeichnet sich u. a. durch seine gute Portabilität aus. UNIX bzw. kompatible Betriebssysteme gibt es für Großcomputer, Minioder Mikrocomputer. International wächst der Kreis der UNIX-Anwender, und mit dem breiten Einsatz der 32-Bit-Technik dürfte seine Rolle weiter wachsen.
Das Buch wendet sich an all jene, die sich einen Überblick über das UNIX-Betriebssystem verschaffen wollen, aber bisher wenig Erfahrungen in dem Umgang mit der Computertechnik haben und auch mit dem einschlägigen Vokabular nicht vertraut sind.
UNIX-Anwendungen werden an Beispielen aus der Büro-Automatisierung (Textverarbeitung, elektronische Post, Computernetzwerke usw.) erläutert. Wichtige charakteristische Merkmale werden soweit beschrieben, wie es für einen Überblick sinnvoll erscheint.
Etwas irreführend bzw. einschränkend wirkt der Titel – denn die Publikation spricht nicht nur „Führungskräfte" an.

I. P.

TGL 44500 Programmiersprache FORTRAN 77

Verlag für Standardisierung, Berlin 1987

Diese Unterlage enthält die Beschreibung der Programmiersprache FORTRAN 77. Es wurde der Standard ISO 1539 – 1980, der ANSI X 3.9 – 1978 entspricht, übernommen.
Die Beschreibung der Programmiersprache FORTRAN 77 erfolgt in englischer Originalfassung, da jede Übersetzung zu beträchtlichen Bedeutungsverschiebungen und Unterschieden in der Realisierung führen kann. Der Standard beschreibt zwei Niveaus der Programmiersprache FORTRAN 77:
1. volles FORTRAN 77 (ISO-Bezeichnung: „Full FORTRAN" oder „FORTRAN")
2. eingeschränktes FORTRAN 77 (ISO-Bezeichnung: „Subset FORTRAN")
Dabei wurde eine interessante Darstellungsform gewählt. Auf den ungeraden Seiten wird der volle und auf den geraden Seiten der eingeschränkte Sprachumfang beschrieben. Zusätzlich sind Unterschiede zwischen beiden Versionen im Text durch senkrechte Striche am Rand gekennzeichnet. Somit ist eine schnelle Orientierung über die im Subset-FORTRAN fehlenden Sprachelemente gegeben.
Der Anhang enthält neben Syntaxdiagrammen Hinweise zu möglichen Portabilitätsproblemen.
Allen Programmierern, die FORTRAN 77 einsetzen, ist dieser 381 Seiten starke Band des Verlages für Standardisierung sehr zu empfehlen.

Ulrich Oefler

Handbuch LSI-Halbleiterspeicher

(Spravočnik Poluprovodnikovye BIS sapominaûščih ustrojstv)
A. Û. Gordonov und Û. N. D'âkov (Herausgeber), in russischer Sprache, Verlag Radio i svâz' Moskau 1986, 360 S.

Dieses Handbuch stellt ein neues Nachschlagewerk zu elektronischen Bauelementen dar, das zusammenfassend in der UdSSR gefertigte LSI-Halbleiterspeicher beschreibt.
Im ersten Teil (Kapitel 1. bis 7.) werden allgemeinbekannte Informationen zu Halbleiterspeichern gegeben (Speicherarten; Ansteuersignale; Kontrolle und Messung; Grundstrukturen der Speicherelemente mit verschiedenen Herstellungstechnologien: TTL, TTL-Schottky, ECL, I²L, I²L-Schottky, CMOS, nMOS; Gehäusetypen; Einbauvorschriften).
Der zweite Teil (Kapitel 8. und 9.) enthält ausführliche Datenangaben zu LSI-Halbleiterspeichern und Anwendungsempfehlungen. Beschrieben werden Speicher-IS der Baureihen

Statische RAMs
TTL: K 134, K 155, KП 185, K 185; **TTL-Schottky:** K 531; **ECL:** K 500, K 1500; **I²L:** K 541, KП 541; **CMOS:** K 176, K 537, K 561; **nMOS:** K 132, K 565.
Dynamische RAMs
nMOS: K 565
ROMs
TTL: K 596; **nMOS:** K 568, K 1610
PROMs
TTL-Schottky: K 556; **ECL:** K 1500
EEPROMs
nMOS: K 573; **pMNOS:** KП 558, KП 1601, K 1601, KM 1609, KП 1609
EPROMs
nMOS: K 573

Das Handbuch sollte für den Praktiker ein nützliches Hilfsmittel sein, gibt es doch einen Überblick über die in der UdSSR verfügbaren LSI-Halbleiterspeicher und vermittelt die notwendigsten Informationen für deren breite Anwendung.

A. Bluschke

Programmierung des 80286

Von C. Vieillefond. Sybex Verlag, Düsseldorf 1987, 1. Auflage, ISBN 3-88745-668-8, 509 S.

Das 16-Bit-Mikroprozessorsystem INTEL 80286 spielt international eine wichtige Rolle beim Einsatz in Personalcomputern (IBM-AT und kompatiblen), Automatisierungssystemen, Terminalarbeitsplätzen u. a. m.
Der vorliegende Titel, es handelt sich dabei um eine Übersetzung aus dem Französischen, ist eine der ersten Buchveröffentlichungen im deutschen Sprachraum, die den Aufbau und die Arbeitsweise dieses Mikroprozessors behandelt.
Nach einer knappen Einführung in den internen Aufbau, die Busstruktur und die Peripheriebausteine wird auf folgende Aspekte bei der Arbeit mit dem 80286 eingegangen:
– Speicherverwaltung und virtuelle Adressierung
Segmente, Adreßräume, Task, Deskriptortabellen
– Speicherschutzmechanismen
Privilegierungsstufen, Gates, Aliassegmente
– Interruptstruktur
NMI-/INTR-Unterbrechungen, Vektortabellen, Exceptions
– Befehlssatz des 80286
Befehlsbeschreibungen
– Programmentwicklungstools
ASM-/BND-/BLD-286
Den größten Umfang nimmt dabei eine sehr ausführliche Beschreibung jedes einzelnen 80286-Befehls im Kapitel *Befehlssatz* ein.
Neben Befehlsformat, Befehlstyp, Beschreibung der betroffenen Bedingungskodes, erfolgt eine genaue Erklärung der Funktionsweise jedes Befehls.
Neben dieser sehr umfangreichen Befehlsbeschreibung kommen die Erläuterungen zu den innovativen Architekturen des 80286 im sogenannten Protected-Mode nach Meinung des Rezensenten viel zu kurz. Dabei sind es doch gerade diese neuen Möglichkeiten, die den Leser interessieren sollten, da sie den Mikroprozessor 80286 gegenüber seinen Vorgängertypen auszeichnen.
Positiv hervorzuheben sind 17 ausführlich dokumentierte Programmbeispiele im Anhang des Buches, die mit der Programmierung des 80286 vertraut machen.

Dr. Ludwig Claßen





1. Berliner Softwarebörse

Vom 3. bis 6. November 1987 fand die erste Berliner Softwarebörse statt, die vom Bezirksneuererzentrum Berlin (BNZ) und der KDT-Anwendergemeinschaft „Rechnergestützte Arbeit“ organisiert und durchgeführt wurde. 73 Betriebe und Einrichtungen der Hauptstadt boten 448 Programmlösungen zur Nachnutzung an. Auf der Börse waren davon 85 Programme im Angebot, die, nach Themengruppen geordnet, von den Programmentwicklern vorgestellt wurden. Wie auch das Bild zeigt, fand diese Veranstaltung ein außerordentliches Interesse. Mehr als 5900 Besucher, davon mehr als 150 Betriebe und Institutionen aus allen Teilen der Republik, nutzten das Angebot. Mehr als 2600 Interessenten trugen sich in die Angebotslisten für die einzelnen Programme ein. Über 1600 Kopien wurden von dem gesamten Softwarekatalog angefordert und übergeben.
Die Softwarebörse hat sich als neue Form für die organisierte Softwarenachnutzung bewährt. Das Vorstellen der Programmlösungen durch den Programmentwickler führte zu einem unmittelbaren Kontakt mit möglichen Anwendern, wobei neben der Vorführung der Programmlösung auch durch unmittelbaren Dialog Entwickler – Anwender alle Fragen geklärt werden konnten. Dieser bewährte Weg muß also fortgesetzt werden.
Während der Softwarebörse haben sich folgende Anwenderlösungen als Spitzenexponate herausgestellt: PERSONAL (VEB Mansfeld Generallieferant), MATERIAL (VEB NILES, STPL (VEB Elektroanlagenbau), RVISION (VEB Applikationszentrum ME), NEUERERW (VEB Wärmeanlagenbau), AMKARTEI (Interflug), UNVINV (VEB Elektro-Apparate-Werke), PLANWT (VEB Kabelwerk Oberspree), Hb Kostenrechnung (RAW Schöneweide) und PRODA (VEB Kosmetik-Kombinat). Diese Lösungen bieten sich auch für eine Nachnutzung in anderen Bezirken der DDR an.
Für die weitere Auswertung der Programme werden mit dem BNZ monatlich zwei „Angebotstage Softwarebörse“ organisiert, wobei jeweils 5 bis 12 Lösungen erläutert und vorgeführt werden sollen (jeweils 9–15 Uhr im BNZ). Die nächsten Termine sind: 13. Mai 1988 (Vermessungsprogramme; Kombinat Kartographie und Geodäsie), 27. Mai 1988 (Erläuterungsberichte und Muster-GAB-Nachweise; Ratioprojekt Berlin) 10. Juni 1988 (Rohrleitungs- und Heizungsberechnungen, Nachweise) und 24. Juni 1988 (Programme der Produktionsvorbereitung).

Dr. J. Schwenke *Foto: Weiß*

3. Klausurberatung UNIX

Vom 10. bis 13. November 1987 fand unter Schirmherrschaft des VEB Leitzentrum für Anwendungsforschung Berlin die 3. Klausurtagung der „Arbeitsgemeinschaft der Entwickler und Nutzer UNIX-kompatibler Software“ (ddr/unix/usr/grp) in Bad Salzungen statt. Diese Arbeitsgemeinschaft vereint seit mehreren Jahren Vertreter von Betrieben und Institutionen der DDR, die auf die Entwicklung und Nutzung von UNIX-kompatibler Software eingeschworen sind (Leitzentrum für Anwendungsforschung Berlin, Technische Hochschule Ilmenau, Technische Universität Karl-Marx-Stadt, Martin-Luther-Universität Halle, ZFT-KEAB Berlin, ZFT-KEAW Berlin, ZKI Berlin, Robotron-Elektronik Dresden, Robotron-Projekt Dresden u. a. m.
Es wurden insbesondere folgende Themenkomplexe behandelt:
1. Stand der Arbeiten an UNIX-kompatiblen Betriebssystemkonzepten und deren Weiterentwicklung
2. Standardisierung der Betriebssystemschnittstellen, Bibliotheksfunktionen, Kommandosprachen und Programmiersprachen
3. Aufbau einer breiten Palette von Anwendungslösungen unter UNIX-kompatiblen Betriebssystemen
4. Verbesserung des Informationsaustausches und der Öffentlichkeitsarbeit zu UNIX-kompatiblen Betriebssystemkonzepten.
Von allen Teilnehmern konnte einheitlich festgestellt werden, daß die Bereitstellung UNIX-kompatibler Betriebssystemlösungen in der DDR in den letzten Jahren große Fortschritte gemacht hat. Das gilt für ESER-Rechner (PSU...) ebenso wie für Kleinrechner (MUTOS 1630...) und für Mikrorechner A 7150 (MUTOS 1700)/P8000 (WEGA).
Von besonderer Bedeutung ist die Tatsache, daß wirklich für **jedes** in der DDR hergestellte Computersystem – gleich welcher Leistungsklasse – der Anwender auf eine leistungsfähige UNIX-kompatible Benutzeroberfläche zurückgreifen kann. Alle Teilnehmer waren sich darin einig, die Arbeiten in den nächsten Jahren konzentriert fortzuführen. Dies gilt besonders für die Erarbeitung von Anwendungslösungen und für die Standardisierungsbemühungen.
Vom VEB LfA wurde die Bereitschaft erklärt, als Träger für eine Nutzergemeinschaft aufzutreten, die ab 1988 insbesondere einer Verbesserung des Informationsaustausches der Anwender UNIX-kompatibler Betriebssysteme dienen soll. **Interessenten wenden sich bitte an den VEB Leitzentrum für Anwendungsforschung, Jacques-Duclos-Str. 47–52, Berlin, 1156, Tel. 5 57 12 45 (Koll. Grützbach).**
Ebenso wird die Redaktion der Mikroprozessortechnik die Bemühungen um eine bessere Öffentlichkeitsarbeit auf dem Gebiet UNIX-kompatibler Software für Mikrorechner unterstützen und geeignete Beiträge und Kurzmitteilungen von Lesern veröffentlichen.

Dr. Ludwig Claßen

Erste Z-1013-Tagung

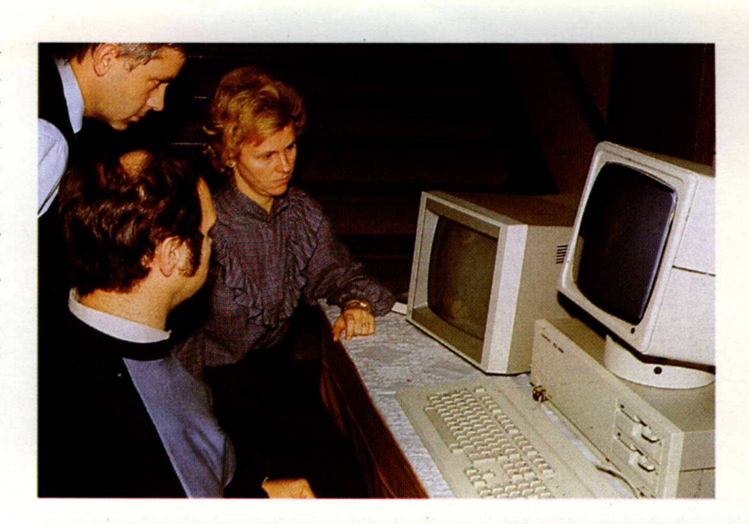
Die *Erste Z-1013-Tagung* wurde am 5. Dezember 1987 von der KDT-Interessengemeinschaft Heimcomputer (IG-HG), Arbeitsgruppe Z 1013 (AG Z 1013), am Informatikzentrum des Hochschulwesens an der Technischen Universität Dresden veranstaltet. Die Tagung sollte Empfehlungen für die Anwender des Mikrorechner-Bausatzes Z 1013 vom VEB Robotron-Elektronik Riesa geben.
Alle Interessenten an dieser Tagung konnten nicht eingeladen werden, da nur 250 Teilnehmer (ca. 1/3 der Meldungen) Platz fanden. Auf Grund der guten Resonanz und der ausgezeichneten Einschätzung wurde eine Wiederholung der Tagung am 27. Februar 1988 vorgesehen.
Als Gäste konnten unter anderem Vertreter der Fachpresse sowie der Abteilungsleiter der Z-1013-Fertigung des VEB Robotron-Elektronik Riesa begrüßt werden. Der Stellvertreter des Direktors für Erziehung aus Ausbildung des Informatikzentrums, *Prof. Dr. H. Löffler*, erläuterte in der Begrüßung die gesellschaftliche Notwendigkeit der privaten Nutzung von HC. Die Eröffnung der Tagung durch den Vorsitzenden der IG HC, *Dr. G. Schönfelder*, brachte zum Ausruck, daß neben der einheitlichen Hard- und Softwareentwicklung auch die kostenlose Weitergabe von Hardwarelösungen und Software zur Nachnutzung an private Z-1013-Besitzer gefordert wird.
Die Referate wurden vom Redakteur der Practic, *R. Besser*, eröffnet, der Regeln für die Dokumentation aufzeigte (sowie Informationen über weitere Tagungen in der DDR gab).
Der Leiter der AG Z 1013, *R. Brosig*, stellte Richtlinien für die kompatible Softwareentwicklung zum Z 1013 vor. Die Schwerpunkte sind: Anwenderprogrammstandorte ab 100H, keine direkten Monitorunterprogrammaufrufe über CALL und keine direkten Tastaturmatrixzugriffe als Nutzerprogramm verwenden, das kommerzielle a „@“ nur für Systemerweiterungen nutzen u. a. m.
Zu Standardisierungsempfehlungen für periphere Schnittstellen nahm *V. Lühne* Stellung.
Mehrere Referenten befaßten sich mit der Schwachstelle des Z 1013, der Tastatur. Nachdem vom Autor ein Überblick zu Tastaturlösungen gegeben wurde, zeigten J. Peters und H. Schuttoff einfache Tastaturlösungen. *R. Brosig* stellte eine komfortable Tastaturlösung (Hard- und Software) vor, mit der nicht nur die Tastencodierung des Z 1013 aus dem Monitor in den RAM-Bereich gelegt werden kann, sondern unter anderem ein Pointer- und Stringfeld frei programmierbar ist.
In der Mittagspause wurde vom VEB Robotron-Vertrieb Berlin Literatur und eine für diese Tagung zusammengestellte Programmkassette angeboten.
Die Tagung wurde von *S. Stölzer* fortgesetzt mit der 64-K-RAM-Erweiterung.
Der Leipziger Computerclub beteiligte sich mit zwei Beiträgen: *Ahrend* stellte die Erweiterungsbaugruppe zum Z 1013 zur EPROM-Programmierung vor. Das Programmsystem FORTH für den Z 10131 erläuterte *Schubert*. Von *A. Köhler* wurde die Druckersteuerung des Z 1013, insbesondere über die V24-Schnittstelle vorgestellt.
Allgemeines Interesse fand die Hardwarelösung eines analogen Joysticks mit einer Auflösung von 1014 Punkten in x- und y-Richtung von *U. Loreit*. Die fertige Leiterplatte wird in diesem Jahr im Handel erhältlich sein.
Der Beitrag „Ladeadressen und Filenamen beim Z 1013“ von *U. Zybel* überforderte die Tagungsteilnehmer, die überwiegend keine Softwareausbildung besaßen.
Ein in der Ag Z 1013 als Standard festgelegtes Kassetteninterface „Headersave“, das zum originalen Kassetteninterface des Z 1013 kompatibel ist und auf der Programmkassette angeboten wurde, erläuterte *R. Brosig*.
Der Abschlußbeitrag von *J. Pohl* befaßte sich mit einem zum Monitor des KC 85/1 fast kompatiblen 4-K-Monitor für den Z 1013.
Der abschließende Softwaretausch dieser Tagung wurde von vielen Teilnehmern genutzt.

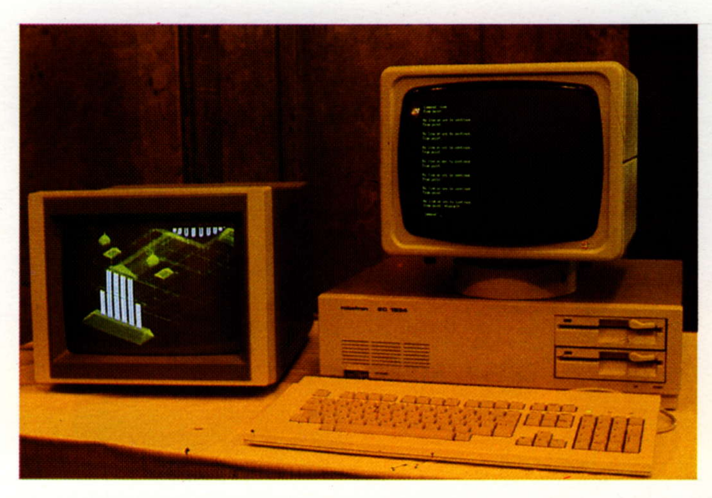
Hans-Ulrich Moik

Fachtagung EC 1834

Am 3. und 4. Dezember 1987 wurde im Karl-Marx-Städter Filmtheater Metropol eine DDR-offene Fachtagung zum Thema *EC 1834 – ein professioneller ESER-Personalcomputer des VEB Kombinat Robotron* durchgeführt. Veranstalter waren die Betriebssektion der KDT des VEB Robotron-Buchungsmaschinenwerk Karl-Marx-Stadt und des VEB Robotron-Elektronik Dresden, Fachgebiet Geräte Karl-Marx-Stadt, sowie der Bezirksvorstand der KDT Karl-Marx-Stadt.
Die Ankündigung der Tagung hatte ein überaus großes Interesse gefunden. Die Zahl der Anmeldungen konnte nicht befriedigt werden, so daß für Januar 1988 sofort eine Wiederholung der Tagung anberaumt werden mußte.
Die Tagung wurde mit einem Vortrag des Betriebsdirektors des VEB Robotron-Buchungsmaschinenwerk Karl-Marx-Stadt, Gen. Dieter Gertler, vor über 450 Teilnehmern im vollbesetzten Saal des Filmtheaters Metropol eröffnet. Er umriß die gesamtgesellschaftliche Zielstellung, die mit der Entwicklung des EC 1834 verbunden ist und erläuterte die Einordnung in das Einheitliche System der elektronischen Rechentechnik (ESER) der RGW-Mitgliedsländer. Er ging davon aus, daß im Fünfjahrplanzeitraum 1986–1990 durch den VEB Kombinat Robotron mindestens 170 000 Büro- und Personalcomputer der Volkswirtschaft bzw. für den Export bereitzustellen sind, verbunden mit einem umfangreichen Spektrum moderner peripherer Geräte zur Systemkomplettierung, insbesondere Drucker, alphanumerische und grafische Bildschirmtechnik, grafische Eingabe- und Ausgabegeräte (Digitalisiergeräte und Plotter) sowie externe Speichertechnik (Disketten- und Festplattenspeicher).
Ein entscheidender Beitrag zur Erfüllung dieser anspruchsvollen Aufgabe wird mit der bevorstehenden Überleitung des EC 1834 in die Produktion geleistet. Wesentlich ist dabei die gemeinsame Produkion dieses Rechners im VEB Robotron-Buchungsmaschinenwerk Karl-Marx-Stadt und im VEB Robotron-Büromaschinenwerk „Ernst Thälmann" Sömmerda auf der Grundlage einer einheitlichen technischen Politik und daraus abgeleiteter rationeller Kooperationsbeziehungen. Anschließend ging Gen. Gertler auf die neuen Anwendungsmöglichkeiten des EC 1834 ein. Durch seine konsequente Orientierung auf ein offenes System, das universell erweiterbar ist, wurde ein arbeitsplatzorientierter Computer mit angepaßtem Softwarekomplex für alle Bereiche der Volkswirtschaft geschaffen, der insbesondere auch als Grundlage für spezialisierte Meß-, Prüf-, Labor- und ähnliche Arbeitsplätze dienen und als Baugruppensatz in Finalerzeugnissen unterschiedlicher Industrievorhaben einbezogen werden kann. Durch die Nutzungsmöglichkeit grafischer Peripherie und leistungsfähiger Softwarepakete ist der EC 1834 auch hervorragend als CAD-Arbeitsstation geeignet.
Im weiteren Velauf der Tagung wurde die Hardware und Software des EC 1834 ausführlich vorgestellt. Zunächst sprach *D. Wiedemuth* zur funktionellen Konzeption und Kompatibilitätseigenschaften. Nach einer Vorstellung der Hardwarebestandteile einschließlich geplanter Systemerweiterungen sowie einem generellen Überblick über das Hauptbetriebssystem DCP erläuterte er ausführlich die definierten und dem Anwender über die Software zugänglichen Schnittstellen. Dabei zeigte er die Übereinstimmung mit den abgestimmten Operationsprinzipien für professionelle Personalcomputer des ESER. Die Schnittstelle des ROM-BIOS zum Betriebssystem garantiert die Kompatibilität zum de-facto-Industriestandard IBM PC/XT. Anschließend ging er auf die Koppelmöglichkeiten des EC 1834 über serielle Kanäle, zu ESER-Systemen über den KIF-Adapter und zur LAN-Kopplung ein.
Dr. Schönyan erläuterte in seinem Vortrag detailliert die Funktionskomplexe der Systemplatine. Besonderes Augenmerk richtete er auf die Anwenderschnittstelle für eigene Hardwareerweiterungen, den Systembus. *S. Radestock* befaßte sich in einem interessanten Vortrag mit der Einbindung der zeichenorientierten Peripheriegeräte Tastatur, alphanumerischer Bildschirm und Drucker, über welche die Mensch-Maschine-Kommunikation realisiert wird. Er stellte die Möglichkeit ladbarer Zeichensätze heraus, welche als Hardwarevoraussetzung für die Mehrsprachigkeit des EC 1834 unerläßlich ist. In weiteren vier Vorträgen wurden Standardadapter bzw. Möglichkeiten für Systemerweiterungen vorgestellt. *F. Espig* behandelte die Einbindung von Festplatte und Floppy-Disk. Der leistungsfähige Farbgrafikadapter wurde mit seinen Parametern und dem Blockschaltbild von *F. Hornig* beschrieben. Zur Realisierung serieller Standardschnittstellen über den ASK-Adapter wurde von *D. Mludek* vorgetragen. Im letzten Vortrag des ersten Tages beschäftigte sich noch einmal *Dr. Schönyan* mit der Ankopplung des EC 1834 an eine Gerätesteuereinheit des Bildschirmsystems EC 7920 über den KIF-Adapter.
Der zweite Tag war ausschließlich Softwareproblemen und Vertriebsbedingungen gewidmet. In einem Grundsatzvortrag dazu ging *M. Philipp* zunächst auf kommerzielle Probleme ein. Insbesondere seine Ausführungen zu Vertriebsbedingungen und -wegen, zu Dokumentation und Schulungsleistungen stießen auf breites Interesse. Anschließend beschäftigte er sich mit Fragen der System- und Datenkompatibilität, insbesondere in Richtung A 7150 und schon vorhandener 8-Bit-Rechentechnik, wie PC 1715 und Bürocomputer A 5120/A5130. Er erläuterte die Möglichkeiten zur Datenkonvertierung von den verbreiteten 8-Bit-Betriebssystem SIOS, UDOS, SCP zu DDP und umgekehrt einschließlich der erforderlichen Hardwarebedingungen. Im nächsten Vortrag stellte *H.-G. Unterschütz* den Kern des Betriebssystem DCP im Detail vor und erläuterte die Werkzeuge der Assemblertechnologie. In einem weiteren Vortrag von *M. Philipp* wurde eine Übersicht zu den angebotenen online-Komponenten und möglichen LAN-Lösungen vorgestellt. *Dr. Männel* beleuchtete die Unterstützung des Personalcomputers EC 1834 in den Betriebssystemen von ESER-EDVA und erläuterte die Emulationsvarianten. *Dr. Köhler* stellte das CAD-Paket MULTICAD für den EC 1834 vor. Er zeigte an einer Reihe von Beispielen die Leistungsfähigkeit dieses Softwarepaketes im Zusammenhang mit der Hardware. Im letzten Vortrag stellte *D. Trinks* das Sprachkonzept des EC 1834 vor. Er verwies darauf, daß alle modernen Programmiersprachen dem Anwender des EC 1834 zur Verfügung stehen bzw. in naher Zukunft verfügbar sind.
Am zweiten Tag wurden vor Beginn der Tagung im Foyer des Metropol drei Personalcomputer EC 1834 in unterschiedlichen Konfigurationen aufgebaut. An ihnen wurde bis zum Nachmittag die Leistungsfähigkeit der vorgestellten Hardware und Software demonstriert (Bild 1). An einer mit Farbgrafik ausgerüsteten Anlage wurde MULTICAD vorgeführt (Bild 2). Die Tagungsteilnehmer konnten sich während dieses gesamten Tages mit allen Softwarepaketen vertraut machen und sich von der Zuverlässigkeit der Geräte überzeugen. Auch ein Blick ins Innenleben der Geräte war erlaubt. Die Bediener hatten alle Hände voll zu tun, um die zahlreichen Demonstrationswünsche zu befriedigen und die vielen Fragen zu beantworten (Bild 3).
Als Fazit der Tagung konnte konstatiert werden, daß den Teilnehmern umfassende Informationen aus erster Hand zu Aufbau, Wirkungsweise und Leistungsfähigkeit des Personalcomputers EC 1834 vermittelt wurden. Insbesondere die am zweiten Tag durchgeführte Demonstration von drei Geräten wurde als besonders wertvoll eingeschätzt. Die Organisatoren der Tagung sind sich einig, auch in Zukunft ihre Neuentwicklung den Anwendern in der DDR mit weiteren Fachtagungen vorzustellen.

Dr. Volkmar Köhler







Fotos: Weiß (3)

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 4
S. 97–128



Chapter 1

The Adventure Begins

This trip really began in September last year when Gerry won first prize in a raffle at the fashion show which Rush-Presbyterian-St. Luke's Medical Center holds every year. The prize was two round trip tickets to Hong Kong on United Airlines, and ten nights in the Hong Kong Hyatt Hotel. Analyzing our good fortune, we concluded that we wanted to do more than spend ten days in Hong Kong and return, but at the same time, United, having just gotten its routes and equipment from Pan American, had not yet received authority to fly to other destinations or between points in the Far East.

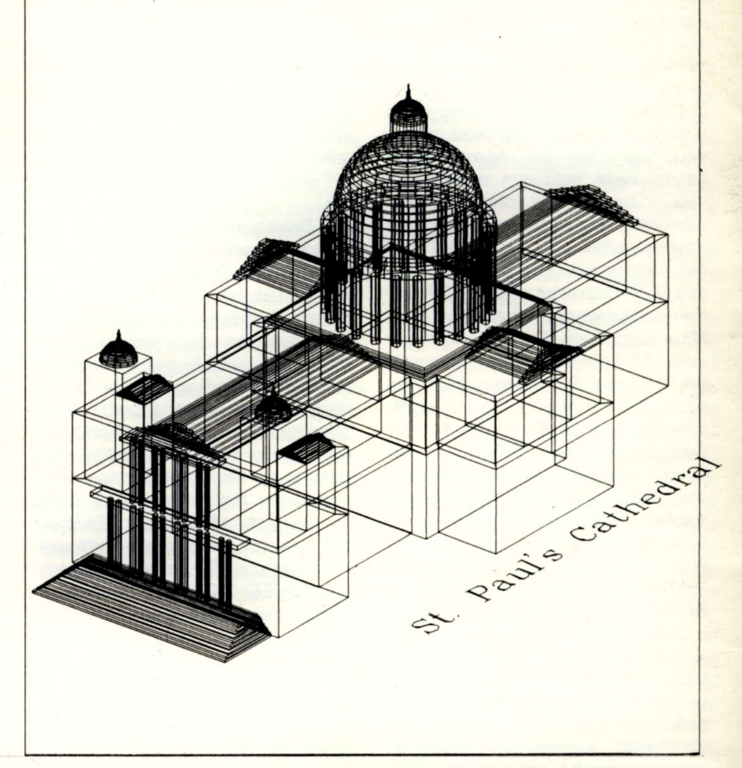
On February 10th, United acquired this authority, and on February 11th Debbie began putting our trip together. We left on March 2nd and returned on March 26th. We entered seven countries, traveled over 25,000 miles on four airlines, made over 500 Kodachrome® slides, almost 200 Kodacolor® prints, and 5 1/2 hours of color and sound videotape.

Chicago to Tokyo

11:03 P.M. Chicago time. 39,000 feet somewhere over the Western Pacific, we are 8 hours and 42 minutes out of Los Angeles with



Figure 1-1. The caption



Was ist Desktop Publishing (DTP)?
Was verbirgt sich hinter dem Kürzel DTP?

Eine Zeitschrift der BRD schrieb darüber: „Dazu muß man zunächst wissen, daß der in letzter Zeit arg gebeutelte Computermarkt händeringend eine neue Kuh benötigt, die gemolken werden kann: Neben rückläufigen Produktions- und Beschäftigungszahlen und einer schleppenden Investitionsneigung steht die Datenwirtschaft in diesem Jahr (1987, d. R.) mit dem Rücken zur Wand. Dollarverfall (auf dessen Basis unsere Exporte abgerechnet werden) und die Billigkonkurrenz tragen zumindest die Teilschuld dieser Entwicklung, der nun mit Hilfe der neuen Boom-Branche DTP entgegengearbeitet werden soll ."

Ohne Zweifel bringt Desktop Publishing für den Anwender dennoch neue Möglichkeiten – nämlich Satz und Layout von Texten auf dem PC. Dabei ist das, was unter DTP oftmals verstanden wird, breit gefächert. Als unterste Stufe könnte man dazu schon einen PC mit Matrixdrucker und modernem Textverarbeitungsprogramm zählen. DTP in „höchster Vollendung" besteht aus PC, komfortablem Layoutprogramm und als Ausgabegerät dient ein Laserbelichter (Fotosatzanlage).

Die Regel liegt nun dazwischen. Voraussetzungen sind ein grafikfähiger PC, ein Laserdrucker und ein Layoutprogramm. Die bekanntesten Layoutprogramme sind *Pagemaker* von der US-amerikanischen Firma Aldus und *Ventura Publisher* von RANK XEROX. Einer der Vorzüge des DTP ist, daß der Anwender den auszugebenden, gestalteten Text so sieht, wie er im Druck erscheinen wird (ohne Steuerzeichen); auch bezeichnet als WYSIWYG – Abkürzung für „What jou see is what jou get". Bei herkömmlichen Fotosatzterminals dagegen ist der Text z. B. durchsetzt mit vielen Steuerzeichen.

Welche Hardware ist erforderlich?

Das zu verwendende System ist natürlich abhängig vom Satzprogramm bzw. gilt es auch umgekehrt.

Der *Ventura Publisher* benötigt z. B. folgende Konfiguration:

- IBM PC XT oder AT oder kompatible mit mindestens 512 KByte Hauptspeicherkapazität, besser sind 640 KByte
- Festplatte mit mindestens 10 MByte (Speicherkapazität)
- Grafikkarte und geeigneter Bildschirm (z. B. 640 × 400 Punkte Monochrombildschirm)
- Maus
- Laserdrucker, falls keine großen Forderungen an die Ausgabequalität stehen, sind auch die Drukker Epson MX-80, FX-80 oder entsprechende ausreichend.

Als Betriebssystem (BS) muß MS/DOS ab Version 2.1 oder höher bzw. ein kompatibles BS verwendet werden.

Zuerst wird der zu verarbeitende Text mit einem Textverarbeitungsystem erfaßt. Der Ventura Publisher liest die Formate der meisten verbreiteten internationalen Textverarbeitungssysteme, wie MS Word, Wordstar oder ASCII-Dateien, wobei Textformatierungen, wie Fettdruck und Unterstreichungen, beibehalten werden. Das Layoutprogramm enthält auch Textverarbeitungsfunktionen, mit denen Texte hinzugefügt oder mittels „Anklicken" mit der Maus u. a. Schriftart, Schriftgröße, Flattersatz (z. B. wie der Rubrikenteil in MP) oder Blocksatz (wie der MP-Aufsatzteil) gewählt werden. Sollen Bilder oder Grafiken, die vorher mittels Scanner oder Grafikprogramm erzeugt wurden, eingefügt werden, zeichnet man auf der gewünschten Seite einen Rahmen, um den der Text automatisch „herumfließt". Nachdem Text, Bilder und Grafiken aus Sicht des Nutzers fehlerfrei sowie richtig positioniert und gestaltet sind, kann die Ausgabe beginnen; sie erfolgt seitenweise. Als Ausgabegerät wird meistens ein Laserdrucker mit einer Auflösung von 300 dots per inch (dpi, Punkte pro Zoll) verwendet. Ist eine höhere Qualität geboten, kann auch auf Laserbelichter ausgegeben werden; z. B. leistet die Linotronic 300 2540 dpi Auflösung.

Dabei steht offensichtlich die Frage nach der (n) entsprechenden Schnittstelle(n) (Kompatibilität).

Besonders preiswerte Systeme erlauben meist nicht die Ausgabe der vom Layoutprogramm erzeugten Anweisungen auf den Laserdrucker oder das Fotosatzgerät eines anderen Herstellers. Das Layoutprogramm „versieht" nämlich den auszugebenden Text mit den nötigen Anweisungen in einer sogenannten Seitenbeschreibungssprache (oder auch Layoutsprache), damit der Laserdrucker „weiß", wie und wo der Text gedruckt werden muß. Der letzte Schritt ist die Erzeugung des Rasterbildes durch den Raster Image Prozessor (RIP). Der RIP befindet sich in der Regel im Drucker, er verwandelt die Beschreibung von Zeichen in die entsprechenden Rasterpunkte und plaziert sie an der richtigen Stelle. Bei einem typischen Laserdrucker mit einer Auflösung von zwölf Rasterpunkten pro Millimeter (300 dpi) besteht eine A4-Seite aus mehreren Millionen Punkten. So dauert es meist eine „ganze Weile", ehe der Drucker die Seite ausgibt, Kopien gehen natürlich schneller. Es gibt derzeit 5 bekanntere Seitenbeschreibungssprachen. Welche sich als „Industriestandard" durchsetzen wird, ist noch ungewiß. Einiges spricht für Postscript von der Fa. Adote, da IBM seine mittleren DTP-Systeme damit versieht. Poscript ist eine stackorientierte Programmiersprache wie FORTH.

Wofür DTP?

In erster Linie ist DTP gut geeignet zum Herstellen von Druck- oder Kopiervorlagen für Handbücher, Dokumentationen, Bedienungsanleitungen, Merkblättern – eben für Dokumente, die für einen größeren Personenkreis sein sollen. Dabei läßt sich über DTP durchaus eine ansprechende Gestaltung erreichen. Für eine perfekte Typografie (u. a. Werbebroschüren, Zeitschriften, Bücher) erfüllt DTP heute noch nicht die Forderungen. Im Verlagswesen für Zeitschriften und Bücher geringerer Auflagen oder in Bereichen, wo es aus bestimmten Gründen nicht auf eine perfekte Gestaltung ankommt, werden zunehmend DTP-Systeme eingesetzt. Eine weitere Hürde sind allerdings die oftmals in den Layoutprogrammen nicht vorhandenen oder schlechten Silbentrennungsprogramme. Abschließend sei noch darauf hingewiesen, daß im Anschluß an den Arbeitsgang über das DTP-System noch das Vervielfältigen (Kopieren oder Druck) erfolgen muß, da Laserdrucker für eine höhere Kopienzahl zu langsam und zu teuer sind.

I.P.

ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK

MP

Heft 5 · 1988

Mikroprozessortechnik

VEB Verlag Technik Berlin

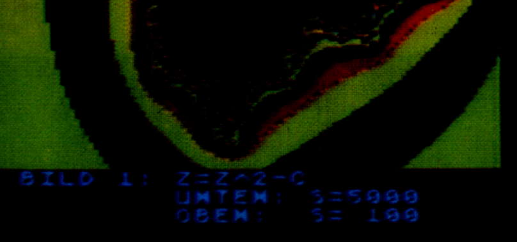
ISSN 0233-2892

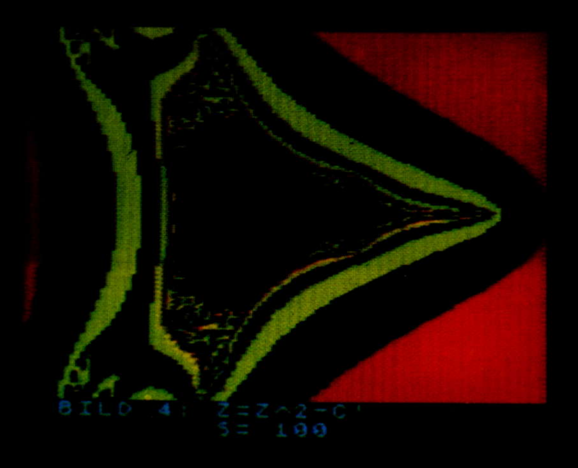


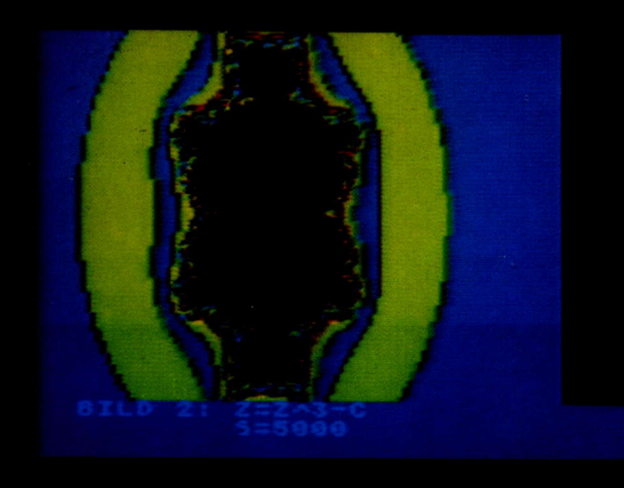
Diagrammdarstellung auf PC 1715

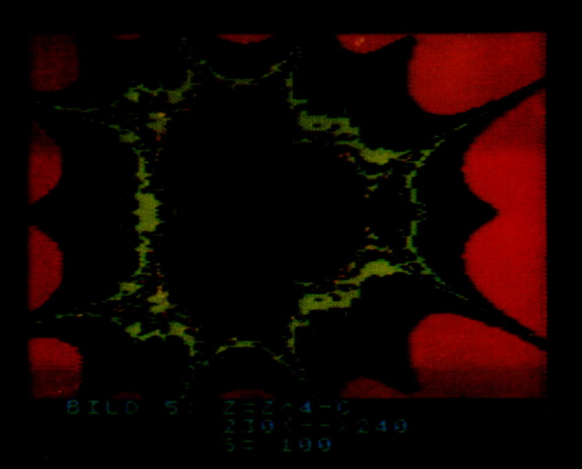
Bustreiberaufsatz für den KC 85/3

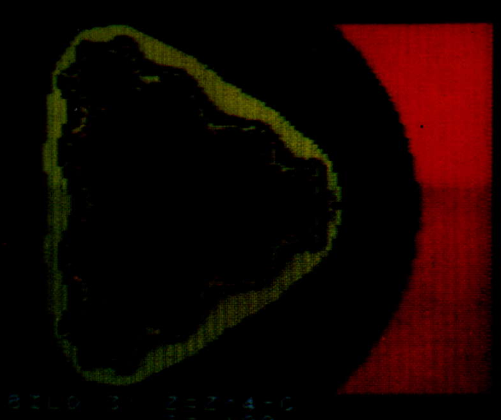










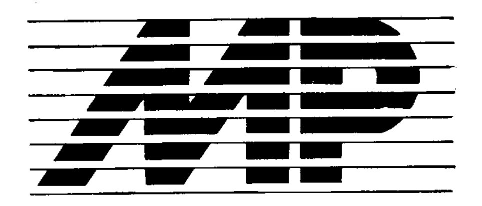




Verhalten von Zahlenfolgen grafisch dargestellt

Lesen Sie dazu den Beitrag von Detlef Thielsch in diesem Heft.

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 5 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 2 87 00, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2 87 03 71); Herbert Hemke, Redakteur (Tel. 2 87 02 03); Sekretariat Tel. 2 87 03 81

Gestaltung Christina Bauer

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

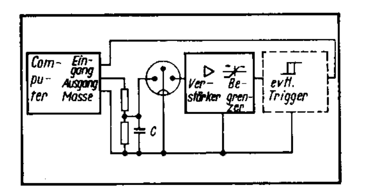
Redaktionsschluß: 15. März 1988

AN (EDV) 49837

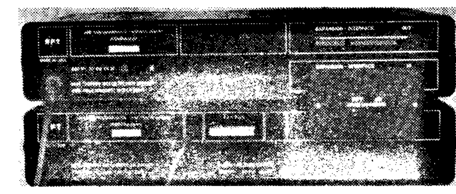
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tísku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Há Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



Seite 136



Seite 149

Zum Titelbild:

Obwohl die Transputertechnik international bisher nur geringe Verbreitung gefunden hat, soll mit unserem Beitrag auf Seite 131 eine Entwicklungsrichtung vorgestellt werden, die für ausgewählte Bereiche der Mikroprozessortechnik in den nächsten Jahren an Bedeutung gewinnen könnte. Ähnlich den vom Transistor gesteuerten elektrischen Strömen liefert der „Computer-Transistor" (Transputer) computergesteuerte Datenströme. Der Beitrag geht auf die Transputergrundkonzeption an Hand des Transputers IMS T424 ein. Nach einer Einführung zum gegenwärtigen Stand und zu Trends bei Computern wird der Transputerbaustein T424 näher vorgestellt. Das Zusammenschalten von Transputern wird unter Ausnutzung einer unterschiedlichen Anzahl von Verbindungsleitungen dargestellt. Weiterhin wird auf die Anwendung des Transputers und die höhere Programmiersprache OCCAM eingegangen.

Vorschau
In MP 6/88 haben wir für Sie unter anderem Beiträge zu folgenden Themen vorbereitet:
- Fehlerorientierte Mikrorechnersysteme
- Rechnerinterface zur Eingabe und Korrektur von Sensordaten
- Sequentielle Online-Verarbeitung von dBase-II-Dateien mit TURBO-PASCAL
- Änderungen am SCP 1700 des AC 1700
- Lichtwellenleiter kontra CSMA/CO?

Inhalt

Neue Schaltkreise in hohen Stückzahlen

Der Stammbetrieb des Kombinates Mikroelektronik steht in diesem Jahr vor der Aufgabe, die Fertigung unipolarer Festkörperschaltkreise gegenüber 1987 auf 180 Prozent zu steigern. An der Entwicklung neuer Generationen wird gearbeitet. Über die Leistungsfähigkeit der modernsten Chipfabrik der Republik, die im vergangenen Jahr sechs Monate früher als geplant übergeben worden war, informierten sich Anfang Februar die Mitglieder und Kandidaten des Politbüros des ZK der SED Günter Mittag, Hans-Joachim Böhme, Werner Eberlein, Siegfried Lorenz, Günter Schabowski, Margarete Müller und Werner Walde. Generaldirektor Prof. Dr. Heinz Wedler berichtete, daß der Januarplan erfüllt wurde und erläuterte die fruchtbare Zusammenarbeit zwischen Kombinat und Territorium. Junge Arbeiter berichteten, sie hätten der Volkswirtschaft im vergangenen Jahr 1,4 Millionen Chips mehr als geplant zur Verfügung gestellt. Während dieser Begegnung mit Vertretern der 34 Jugendbrigaden und 24 Jugendforscherkollektive des Stammbetriebes sagte Günter Mittag unter anderem, die neue Produktionsstätte für den 64-Kilobit-Speicherschaltkreis in Erfurt-Südost zeige, wie nach den Beschlüssen des XI. Parteitages der SED ein entscheidender Schritt getan werde, um 1988 leistungsfähige Schaltkreise in Massenproduktion für die Volkswirtschaft zur Verfügung zu stellen. Er würdigte den Anteil der Erfurter Mikroelektroniker am Durchbrechen imperialistischer Embargos.

ADN

Elektronisierung, Computer und Zusammenarbeit im RGW

Unter den Hauptrichtungen des wissenschaftlich-technischen Fortschritts der RGW-Länder bis zum Jahre 2000 nimmt die Elektronisierung einen besonderen Platz ein. Die Zusammenarbeit auf diesem Gebiet umfaßt 35 Themen. Ihr Spektrum ist recht weit gefächert – es reicht von der gemeinsamen Forschung, Entwicklung, Produktion und Anwendung von Rechentechnik bis hin zum Gerätebau, zu Lasertechnik und Nachrichtenmitteln. Bezeichnend ist unter anderem das Beispiel des Kombinats Robotron. Bei Orientierung auf den sozialistischen Markt (70 Prozent der Erzeugnisse gehen in RGW-Länder) erzeugt dieser Betrieb eine breite Palette: Elektronische Schreibmaschinen und Personalcomputer, Terminals und Steuerrechner arbeiten zuverlässig in den RGW-Ländern. Ein hohes technisches Niveau bei der Produktion komplizierter elektronischer Erzeugnisse haben auch Videoton (UVR) und Tesla (ČSSR). Durch die Kooperationsbeziehungen wächst und erneuert sich ständig das Sortiment elektronischer Bauelemente. Gestützt auf langjährige Erfahrungen haben die RGW-Länder jetzt eine neue, höhere Stufe der Zusammenarbeit auf der Grundlage des Komplexprogramms des wissenschaftlich-technischen Fortschritts erreicht. In den letzten zwei bis drei Jahren ist die Serienproduktion vieler neuer Maschinen in der UdSSR und der DDR aufgenommen worden. In der ČSSR, der VRB und der DDR wird angestrebt, demnächst neue moderne Rechner verschiedener Leistungsklassen mit bis zu zehn Millionen Operationen pro Sekunde zu entwickeln. Polen entwickelt dafür vor allem Technologien und Ausrüstungen zur Herstellung mehrschichtiger Leiterplatten, grafische sowie numerische und alphanumerische Anzeigen, Ungarn spezialisiert sich unter anderem auf logische Informations- und Rechnersysteme. Ab 1988 beteiligen sich die rumänischen Spezialisten aktiver an der Zusammenarbeit. Sie wollen wichtige Bauelemente sowie grafische Arbeitsstationen liefern.
Die begonnene industriemäßige Züchtung von Silizium-Einkristallen mit einem Durchmesser bis 200 Millimeter und einer Masse von 60 Kilogramm ist eine wichtige Voraussetzung für die Herstellung von Chips. Die Wissenschaftler des bulgarischen Instituts für Festkörper und des sowjetischen physikalisch-technischen Joffe-Instituts konstruierten eine Spezialvorrichtung, durch die man Trägerschichten mikroelektronischer Schaltkreise mit idealer Oberfläche erhält. Nun wird in Plowdiw ein internationales Laboratorium zur Entwicklung dieser Technologie und ihrer Einführung in die Produktion geschaffen. Bekanntlich wurde in Kooperation zwischen Betrieben der UdSSR und dem Kombinat Carl Zeiss Jena mit der Herstellung technologischer Ausrüstungen zur Produktion integrierter mikroelektronischer Schaltkreise mit 1 MBit Speicherkapazität begonnen, was dem Weltniveau entspricht. So setzt sich immer stärker die Tendenz zu einem Niveau kooperativen Zusammenwirkens durch, das es gestattet, in sämtlichen Abschnitten des Prozesses „Wissenschaft – Technik – Produktion – Absatz" den geplanten und erwarteten technologischen Schub zu realisieren.

ADN-Medwedew

China verstärkt Produktion von Computern

In den vergangenen zwei Jahren wurden in Peking viele Elektronikläden eröffnet.
Sie bieten sowohl der Industrie und Verwaltung, aber auch privaten Interessenten eine breite Palette, vom Kleincomputer bis zum vollfarbgrafikfähigen 16-Bit-Personalcomputer.

Chinas erster Rechner war 1958 entstanden. Aber erst mit Beginn der Wirtschaftsreform und der Öffnung in den 80er Jahren setzte auf der Grundlage von Importen die Entwicklung einer eigenen Schaltkreisindustrie ein.
Auf der Basis von IBM-Standards und unter Verwendung importierter Komponenten begann in China auch die Produktion von 16-Bit-Personalcomputern und Peripherie. Rund 150 000 dieser Rechner, ein Großteil davon Importe, sind heute im Einsatz. Ziel ist es, zunehmend Schaltkreise aus eigener Produktion zu verwenden. Zahlreiche Varianten wurden inzwischen für die Eingabe von chinesischen Schriftzeichen in Computer geschaffen.
Der akute Mangel an hochqualifiziertem Personal, insbesondere im Software-Bereich, ist nach chinesischer Einschätzung noch ein Hindernis für die weitere Entwicklung dieses Zweiges. Verstärkte Ausbildung soll in den kommenden Jahren Abhilfe schaffen.

ADN/MP

Ingenieurbetriebe helfen

Die 15 Ingenieurbetriebe für die Anwendung der Mikroelektronik unterstützten im Vorjahr insbesondere Klein- und Mittelbetriebe mit 159 speziellen wissenschaftlich-technischen Leistungen, deren Nutzen sich auf über 48 Millionen Mark belief. Damit konnten die Anwender 1987 allein 102 000 Stunden Arbeitszeit einsparen. Die Ingenieurbetriebe, in den letzten beiden Jahren in allen Bezirksstädten der DDR aufgebaut, geben den Betrieben vor allem im Rahmen der territorialen Rationalisierung aktive Hilfe beim Einsatz der Mikroelektronik und Rechentechnik. Derzeit vereinen sie etwa 600 hochqualifizierte Kader, die über moderne rechentechnische Mittel zum Entwurf spezieller Gerätetechnik und von Software verfügen. Sie bieten vor allem Erstanwendern der Mikroelektronik einen umfangreichen Service. So wirken die Ingenieurbetriebe beim Entwurf und der Applikation anwenderspezifischer Schaltkreise mit und bieten als Dienstleistung ihren Partnern Schulung und Einweisung zum Bedienen, Warten und Instandsetzen mikroelektronischer Technik. Im Vorjahr hat sich der ökonomische Nutzen dieser Gemeinschaftsarbeit für die Anwender gegenüber 1986 nahezu verdoppelt. Durch Fachvorträge und Konsultationen unterstützen die Ingenieurbetriebe zudem Erstanwender der Mikroelektronik und Mikrorechentechnik aktiv bei der Qualifizierung. So wurden 1987 fast 18 000 Werktätige aus Klein- und Mittelbetrieben der DDR in 513 Lehrgängen mit moderner Rechentechnik vertraut gemacht. Deshalb wird die Zusammenarbeit der Ingenieurbetriebe mit örtlichen Staatsorganen, insbesondere den Bezirkswirtschaftsräten, ausgebaut.

ADN

Unix-Mehrplatzgrafik

Als Neuheit auf dem Unix-Markt bietet Siemens jetzt Mehrplatzgrafik in seinen Sinix-Computersystemen an. Damit können erstmals die Vorteile von Grafikarbeitsplätzen wie einfachste Bedienung und vergrößerter Funktionsumfang (objektorientierte Bedienoberfläche und Fenstertechnik) nicht nur an einem, sondern an generell allen Arbeitsplätzen eines Mehrplatzsystems genutzt werden. So kann der gleichzeitige Ablauf von mehreren Programmen (Multitasking) in mehreren Fenstern auf dem Bildschirm verfolgt werden. Basis dafür ist der neue Grafik-Bildschirmarbeitsplatz 97808, ausgerüstet mit eigenem Prozessor und eigenem Speicher. Er kann sowohl in der Grafik-Betriebsart als auch in der üblichen alphanumerischen Betriebsart arbeiten. Sein Schwarzweißbildschirm mit 15 Zoll Diagonale hat eine Bruttoauflösung von 792 × 594 Pixel. Alle Pixel sind einzeln ansteuerbar. Die hohe Bildwiederholfrequenz von 70 Hz sorgt dabei für eine flimmerfreie Darstellung.

MP

In eigener Sache

Für eine weitere Redakteurstelle in unserer Zeitschrift suchen wir einen geeigneten Mitarbeiter bzw. eine Mitarbeiterin mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik.
Zu den Aufgaben gehören unter anderem das redaktionelle und fachliche Bearbeiten von Manuskripten, der Besuch und die Auswertung von Fachtagungen, -messen und -ausstellungen, die Zusammenarbeit mit Autoren und Gutachtern sowie ggf. das Testen und Beurteilen von Programmen, die der Redaktion zur Veröffentlichung eingereicht werden.
Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, rufen Sie uns unter Tel. 2870371 oder 2870203 an oder schreiben Sie an:
VEB Verlag Technik, Redaktion MP, Oranienburger Str. 13/14, Berlin, 1020

Transputer

Martin Sattelkau
Ingenieurhochschule für Seefahrt Warnemünde/Wustrow

Ziel des Beitrages ist es, die Transputergrundkonzeption an Hand des Transputers IMS T424 und die Sprache OCCAM vorzustellen. Als Einheit stellen sie eine Möglichkeit dar, die Leistungsfähigkeit von Datenübermittlungsanlagen zu erhöhen.

Mit dem Auftreten immer komplexerer Problemstellungen in allen Bereichen des Lebens, besonders in Wissenschaft und Technik, ergibt sich die Notwendigkeit, die Verarbeitungsgeschwindigkeit von Computern stark zu erhöhen. Ein Weg besteht darin, die Fortschritte bei der Erhöhung der Rechnerleistung durch Verbesserungen in der Bauelemente-Technologie zu erzielen. Jedoch nähert man sich hierbei einer physikalischen bzw. technologischen Grenze. Dies gilt besonders für die wichtige technische Kennziffer, die Operationsgeschwindigkeit, die zwar nicht das einzige Maß für die Leistungsfähigkeit eines Computers darstellt, aber trotzdem als ein fundamentales Merkmal angesehen werden muß /1/. Ein anderer Weg bestand und besteht in der Weiterentwicklung der konventionellen J.-v.-Neumann-Architektur. Die Verbesserung der Software war und ist ebenfalls ein Weg zur Erhöhung der Leistungsfähigkeit bestehender und zukünftiger Rechner. In der Gegenwart wird versucht, den geforderten Leistungszuwachs mit Hilfe neuer Rechnerarchitekturen, die z. B. durch die Parallelverarbeitung von Daten gekennzeichnet sind, zu erzielen. Für die Erhöhung der Verarbeitungsgeschwindigkeit durch teilweise oder vollständige Parallelverarbeitung spricht – unter anderem –, daß man schon gegenwärtig mit der zur Verfügung stehenden Technik an die Lösung dieser Aufgabenstellung gehen kann /1/, /2/. Jedoch werden schließlich nur jene Rechner den notwendigen Leistungszuwachs in der Informationsverarbeitung erzielen, die auf der Basis revolutionärer Technologien und Architekturen arbeiten. Hier wären z. B. die Forschungsrichtungen für Biocomputer /3/ und reine optische Informationsverarbeitungssysteme /4/ zu nennen.

Der Transputer IMS T424

Dem Transputerkonzept /5/, /6/, /7/ liegt die Idee zugrunde, mit Hilfe eines speziellen Mikrorechnerbausteines leistungsfähige Multiprozessorsysteme, die nicht unter dem Problem der zunehmenden Busbelastung bei steigender Anzahl von Prozessen leiden, zu realisieren.
Der Transputerbaustein IMS T424 (Bild 1) wird in CMOS-Technik hergestellt. Er ist in einem 84poligen Chip-Carrier-Gehäuse aus Keramik untergebracht, beherbergt einen 32-Bit-Hochleistungsprozessor in RISC-Architektur (Reduced Instruction Set Computer) mit 48 Befehlen und 50 ns Zykluszeit. Ein statischer 4-KByte-RAM mit 35 ns Zugriffszeit, vier Kommunikationskanäle (Link 0-3), seriell und bidirektional, mit je 1,5 MByte/s Datenrate pro Richtung, ein 32-Bit-Speicher-Interface (25 MByte/s), ein 8-Bit-Peripherie-Interface (4 MByte/s) und ein Zeitgeber (32-Bit mit 1 μs Auflösung) als Echtzeitbasis für zeitabhängige Probleme gewährleisten einen hohen Datentransfer.
Durch diese komplexen Funktionen auf einem Chip ergeben sich verschiedene Vorteile, so unter anderem, daß der Bauelementeaufwand gering ist, ein externer Zeitgeber entfällt und die Abmessungen des Multiprozessorsystems günstiger ausfallen. Ein weiterer Vorteil ist die Komplexität eines jeden Transputers zu anderen Transputergenerationen. Dies wird unter anderem dadurch erleichtert, daß dem Transputer bei der Herstellung eine Signatur eingeschrieben wird, die per Programm gelesen werden kann, wodurch dann Typ, Geschwindigkeit, Wortbreite usw. erkannt werden können. Die CPU enthält acht 32-Bit-Register (Bild 2). Jedes Register führt eine spezifische Funktion aus. Drei der acht Register bilden einen Stapelbereich für arithmetische und andere Datenoperationen. Der Befehlszeiger arbeitet in gewohnter Weise. Der Arbeitsraumzeiger enthält die Basisadresse des Arbeitsspeicherbereiches eines Prozesses. Bei jeder Prozeßumschaltung wird er aktualisiert. Mit den F- und B-Registern wird der Anfang und das Ende einer Warteschlange, sie dient der Überwachung der einzelnen Prozesse, verwaltet. Das Befehlsformat (Bild 3) ist für alle Befehle gleich. Es besteht aus zwei Feldern zu je 4 Bit Operationscode und Operand. Da 4 Bit nur 16 Befehle ergeben, kann der Befehlssatz mit speziellen Präfix-Befehlen, das sind Befehle, die vor anderen Befehlen eingefügt werden, erweitert werden.
Der interne RAM-Bereich kann mit dem vielseitigen Speicher-Interface (Bild 4) erweitert werden, und über das Peripherie-Interface können Peripherie-Bausteine angesteuert werden. Alle Schnittstellen arbeiten unabhängig voneinander und dürfen gleichzeitig angesteuert werden.
Mit Hilfe der vier Kommunikationskanäle ist es möglich, die Transputer-Bausteine zu einem Multiprozessorsystem zusammenzuschalten. Dabei ergeben sich vielfältige Möglichkeiten, dem jeweiligen Problem angepaßte Strukturen zu bilden (Bild 5); auch räumliche Gebilde sind, indem man zwei Transputer zu einem Knoten verbindet (Bild 6), möglich.
Prozessorsysteme lassen sich mit einem Transputer oder mit beliebig vielen errichten, wobei ein OCCAM-Programm, welches für mehrere Prozessoren geschrieben wurde, auch auf einem einzelnen Transputer läuft. Dabei besteht nicht die Notwendigkeit, das ganze Softwarepaket umzuschreiben. Lediglich ein paar Zuweisungen am Beginn des Programms müssen dann geändert werden.
Die Kommunikation zwischen den einzelnen Transputern erfolgt prozessorunabhängig und asynchron. Bild 7 /8/ zeigt das Datenformat des Kommunikations-Protokolls.
Nachteile des IMS T424 sind z. B., daß /9/:
– sich der Datenaustausch auf eine Speicher-Speicher-Kommunikation beschränkt und somit ein Zugriff mehrerer Transputer auf einen gemeinsamen Speicherbereich nicht möglich ist.
– nur Daten zwischen den Speichern direkt über Kommunikationswege, genannt „Links", miteinander verbundener Transputer ausgetauscht werden können. Dies bedeutet, daß ein Durchschalten der Links durch mehrere Transputer zu einem sogenannten Fernweg (telepath) nicht möglich ist.
– sich die räumlichen Transputeranordnungen nur durch das Zusammenschalten von zwei Transputern zu einem Knoten realisieren lassen.
– der interne Speicher für verschiedene Anwendungszwecke zu klein sein wird.
Weitere Aussagen zu Transputern finden sich z. B. in den Quellen /10/, /11/, /12/, /13/.

Anwendungsbeispiele

Probleme, die als parallele Prozesse strukturierbar sind, lassen sich mit Hilfe von Transputernetzwerken leicht bearbeiten /7/. Ein Grund hierfür ist, daß sich die Transputernetzwerke gut an die jeweilige Aufgabenstel-

Bild 1 Transputerbaustein IMS T424

Bild 2 Registersatz der CPU

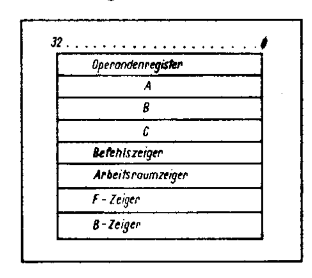


Bild 3 Befehlsformat

Bild 4 Speicherinterface

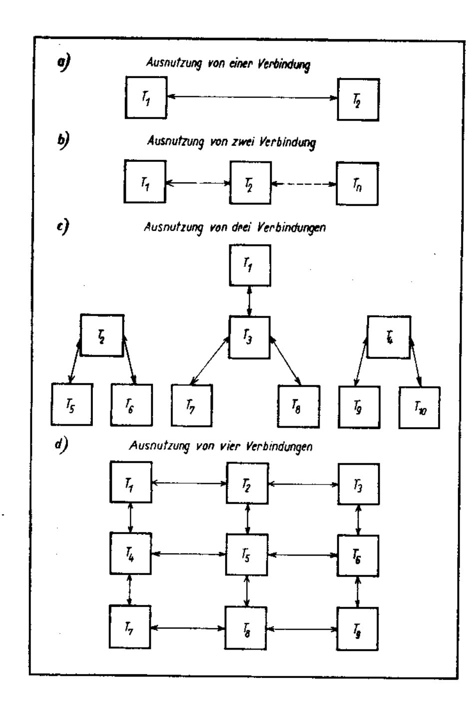
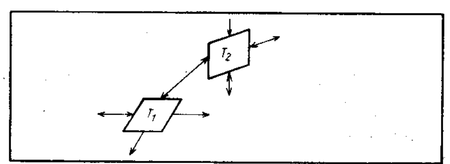


Bild 5 Beispiele für das Zusammenschalten von Transputern unter Ausnutzung einer unterschiedlichen Anzahl von Verbindungsleitungen

Bild 6 Transputerknoten mit sechs Verbindungen zum Aufbau von räumlichen bzw. gemischten Strukturen



lung anpassen lassen. Beispielsweise werden beim Betreiben von Datenbanksystemen oder der Echtzeit-Signalanalyse hohe Datendurchsatzraten benötigt, die sich dann mit einer entsprechenden Anzahl von Transputern realisieren lassen. Als klassisches Beispiel für die Anwendung von Transputern findet man in der Literatur /7/, /14/ die schnelle Fouriertransformation (FFT – Basisalgorithmus der Signalanalyse), da bei der FFT zahlreiche gleichartige Operationen, Schmetterlingsoperationen genannt, durchgeführt werden. Hierbei kann die Struktur der FFT auf ein lineares Feld (Pipeline) von Transputern abgebildet werden. Solch ein lineares Feld kann

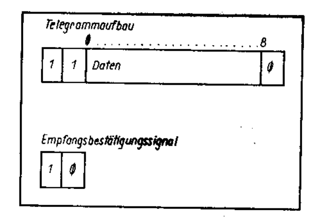
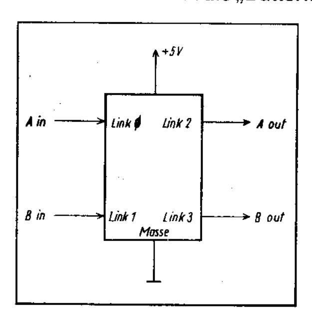


Bild 7 Datenformat des Kommunikations-Protokolls

Bild 8 Der IMS T424 als „Butterfly"-Maschine



aber nur eine bestimmte Abtastfrequenz verkraften. Werden höhere Abtastraten benötigt, können mehrere lineare Transputerfelder parallel verwendet werden. Wird beispielsweise eine Transformation von 64 Meßwerten gewünscht, kann man sechs Transputer in Reihe schalten, und für 1024 Werte können 10 Transputer linear angeordnet werden. Bei der Bildverarbeitung (1024 × 1024 Bildpunkte), die den FFT-Algorithmus auch verwendet, würde man mit einem quadratischen Feld von 10 × 10 Transputern minimale Verarbeitungszeiten erreichen. Bild 8 zeigt den Transputer IMS T424 als „Butterfly"-Maschine.

Folgende zwei Demonstrationsbeispiele für die Leistungsfähigkeit von Transputern wurden in /15/ vorgestellt. In beiden Anwendungsfällen kam der Transputertyp IMS T414, der ebenfalls ein 32-Bit-Hochleistungsprozessor ist /16/, zum Einsatz.

Im ersten Beispiel wird ein System von 16 Transputern, angeordnet als 4 × 4-Matrix, das der Datenreduktion bei der Identifikation von Fingerabdrücken dient, vorgestellt.

Hierbei besteht für das System die Aufgabe, aus etwa 250 KByte Rohdaten rund 1 KByte kodierter Nutzerinformation über die wichtigsten Charakteristika eines Fingerabdrucks zu destillieren. Das System arbeitet diese Aufgabe in rund drei Sekunden ab.

Als zweites Beispiel wird ein System beschrieben, welches einen Fingerabdrucksatz, bestehend aus 10 Einzelabdrücken von jeweils 1 KByte Datenvolumen, mit 3,5 Millionen gespeicherten Fingerabdrucksätzen vergleicht und eine eventuelle Übereinstimmung erkennt. Gelöst wird diese Aufgabe mit einem System von 100 Transputern in quadratischer 10 × 10-Konfiguration und einem Plattenspeicher; es erreicht etwa 500 Vergleiche pro Sekunde.

Programmiersprache OCCAM

OCCAM /7/, /17/ ist eine höhere Programmiersprache (Fachsprache), die das Modell der parallelen kommunizerenden Prozesse unterstützt. Der Transputer setzt dieses Modell in die Hardware um.

OCCAM ist blockorientiert, obwohl es keine „Begins" und „Ends" gibt. Sie wurde für Systeme mit einer beliebigen Anzahl von Prozessoren definiert. Es findet echte Parallelarbeit statt, d. h., Teile des Programms werden gleichzeitig abgearbeitet. Eine Ausnahme ergibt sich beim Betreiben von nur einem Prozessor; hier bearbeitet der Transputer abwechselnd verschiedene Programmteile. Durch diese Konzeption ist es möglich, frei zu entscheiden, wieviel Transputer verwendet werden sollen, da die Softwarepakete (bis auf einige Zuweisungen am Anfang des Programms) nicht verändert werden müssen.

Der Informationsaustausch zwischen den einzelnen OCCAM-Prozessen findet über OCCAM-Kanäle statt. Wenn OCCAM-Prozesse in mehreren Prozessoren gleichzeitig ablaufen, werden die OCCAM-Kanäle durch Punkt-zu-Punkt-Schnittstellen gebildet. Hierzu verfügt der Transputer über die Links.

Die Basis von OCCAM wird durch folgende fünf primitive Prozesse gebildet:

Assignment	– Veränderung des Wertes einer Variablen
Input	– einen Wert von einem Eingabekanal lesen
Output	– einen Wert in einen Ausgabekanal schreiben
SKIP	– keine Operation, beendet aber einen Prozeß
STOP	– bricht einen Prozeß ab.

Tafel 1 Auswahl der OCCAM-Befehle

Sprachelement	Beispiel	Erklärung
Kanal-vereinbarung	CHAN a,b: CHAN c[10]:	'a' und 'b' sind zwei Kanäle. 'c' ist ein Feld von 10 Kanälen.
Prozeß-vereinbarung	PROC d(CHAN a,b)=	Ein Prozeß mit dem Namen 'd' und den zwei Schnittstellenkanälen 'a' und 'b' wird vereinbart.
Variablen-vereinbarung	VAR x,bool.x,dd VAR Vektor[100]	Vereinbarung von 3 Variablen Feld mit 100 Elementen
Einrückungen (um 2 Leer-zeichen)	SEQ r:=0 PAR i=[0 FOR 10]	Einrückungen werden verwendet, um eine Menge von Prozessen einzuklammern.
Kommentar (Bindestrich)		Einem Kommentar muß immer ein Bindestrich vorangestellt werden.
Eingabe	in?x	Lesen eines Wertes vom Kanal 'in'.
Ausgabe	aus!Zeichen.x	Schreiben eines 'Zeichens' und eines Wertes in den Kanal 'aus'.
Zugriff auf ein Byte	Zeichen:=h [BYTEx]	Der Variablen 'Zeichen' wird das x-te Byte des Feldes 'h' zugewiesen.
Bedingung	IF x=0 x:=x+1	herkömmliches Bedingungsstatement Wenn x=0, dann ...
Schleife	WHILE x=0 in?x	herkömmliche WHILE-Schleife Lies solange, bis x≠0 ist.
Zuweisung	x:=x+1	Der Wert von x wird um 1 erhöht.
Sequenz	SEQ in?x aus!x	Sequentieller Block von Prozessen.
Parallelität	PAR i=[0 FOR 10] Vek[i]:=0	10 parallele Prozesse Jeder Prozeß setzt ein Element des Feldes 'Vek' auf den Wert 0
Alternativen	ALT i=[0 FOR 10] eingang[i]?x aus!x	10 gleichberechtigte Prozesse Der erste aktive Kanal und sein Prozeß werden bedient. Der gelesene Wert wird ausgegeben.

Eine Auswahl der OCCAM-Befehle gibt Tafel 1. Zusammenfassend läßt sich feststellen, daß OCCAM eine kleine und kompakte Sprache ist und das einzelne OCCAM-Programm eine Zusammenstellung von OCCAM-Prozessen ist. Der Anwender legt bei dieser Sprache explizit den Grad der Parallelverarbeitung fest.
Weitere Aussagen zu der Sprache OCCAM und kommentierte Programmbeispiele finden sich in den Quellen /18/, /19/, /20/.

Literatur

/1/ Heuer, H.: Parallelverarbeitende Rechnersysteme. Mikroprozessortechnik 1 (1987) 3, S. 71–74
/2/ Kazmierczak, H.: Erfassung und maschinelle Verarbeitung von Bilddaten. Berlin 1980
/3/ Paugstadt, W.: Biocomputer noch nicht in Sicht. Technische Gemeinschaft 35 (1987) 3, S. 24
/4/ Optischer Computer in Sicht? Urania (1987) 11, S. 26
/5/ Eckelmann, P.: Transputer: Mikrorechner-Konzept für hohe Verarbeitungsleistung. Elektronik 32 (1983) 24, S. 51–55
/6/ Eckelmann, P.: Architektur und Anwendung des Transputers. Elektronik 33 (1984) 4, S. 59–65
/7/ Ebert, H.: Ein Transputer kommt selten allein. c't (1985) 1, S. 80–88
/8/ Whitby-Strevens, C.: The Transputer. Computer Architecture Conference Proceedings/New York: IEEE (1985) XIV, S. 292–300
/9/ Endriß, H.; Lawitzky, G.; Oechslein, H.: Moderne Mikroprozessoren. Elektronik 35 (1986) 15, S. 79–84
/10/ Eckelmann, P.: Multiprozessorsysteme als eine logische Konsequenz. Markt & Technik (1984) 46, S. 67–73
/11/ Vajda, F.: Critical Issues of the Application of a Transputer in a Concurrent System. Euromicro Symposium (1985), S. 315–326
/12/ Hellmold, K.-U.: Der Simulationsrechner SIMPLEX. Erlangen 1985
/13/ Schindler, M.: Mikroprozessoren: An der Schwelle zur 2. Computer-Ära. Elektronik 36 (1987) 10, S. 73–80
/14/ Eckelmann, P.: Transputer – richtig eingesetzt. Elektronik 34 (1985) 4, S. 57–62
/15/ Schmidt, E.: Der Transputer wurde von der Software her erdacht. Computerwoche (1986) 15, S. 53–56
/16/ Lineback, J. R.: Parallel Processing: Why a shakeout nears. Electronics (October 28, 1985), S. 32–34
/17/ Eckelmann, P.: Methodisches Programmieren in OCCAM. Elektronik 33 (1984) 21, S. 218–223
/18/ May, D.: OCCAM: Programmiersprache für den Systementwurf. Elektronik 31 (1982) 22, S. 83–86
/19/ Wilson, P.: OCCAM Architecture Erases System Design – Part I. Computer Design 22 (1983) 13, S. 107–115
/20/ Wilson, P.: Language-Based Architecture Erases System Design – II. Computer-Design 22 (1983) 14, S. 109–120

✉ KONTAKT ☏
Ingenieurhochschule für Seefahrt Warnemünde/Wustrow
Wissenschaftsbereich Navigation
Warnemünde 2530

Arbeit mit ASCII-Dateien im Betriebssystem SCP

Kornelia Müller, Michael Lennartz

1. ASCII-Standarddateien

Im Betriebssystem SCP kann auf vielfältige Weise mit ASCII-Dateien gearbeitet werden. Praktisch jedes Programmsystem (Textprozessor, BASIC-80 u. a.) erzeugt ASCII-Dateien. Häufig werden unterschiedliche Programmsysteme miteinander gekoppelt, um spezielle Leistungsparameter effektiv einzusetzen. Solche Verbindungen sind beispielsweise:
– Dateneingabe mit REDABAS unter Nutzung der Eingabemasken
Weiterverarbeitung mit BASIC
– Eingabe und Verarbeitung mit BASIC, Ausgabe sortierter oder mit FOR-Bedingung selektierter Datensätze mit REDABAS
– Umstellung von BASIC auf PASCAL, weitere Nutzung der BASIC-Dateien
– Berechnung mit KP und Auswertung/ Druck mit REDABAS usw.

Unmittelbarer Datenaustausch zwischen einzelnen Programmsystemen ist jedoch nicht ohne weiteres möglich. Standardsoftware arbeitet mit modifizierten Datenformaten, die dem Leistungsvermögen der einzelnen Programme angepaßt sind.
Eine ASCII-Datei besteht ausschließlich aus Zeichen, die im ASCII-Code verschlüsselt sind. Für alphanumerische Informationen ist das die günstigste Darstellungsart. Zahlen hingegen können auf unterschiedliche Weise in ein Hexadezimalformat überführt werden. In einer ASCII-Standard-Datei werden auch Zahlen im ASCII-Code dargestellt. Dadurch ist ein einfacher, von hexadezimalen Umwandlungen oder prozessorinternen Darstellungsarten unabhängiger Übertragungsprozeß von Informationen gewährleistet. In einer Standarddatei werden einzelne Datensätze durch CR LF (ODH OAH) getrennt. Das Ende einer Datei wird mit SUB (1AH) gekennzeichnet. Ein Beispiel zeigt Bild 1. Diese Form der ASCII-Datei garantiert die volle Verarbeitbarkeit einer Datei in allen Standardprogrammen des Betriebssystems. Sie kann mit dem residenten Kommando TYPE aufgelistet bzw. gedruckt werden. In den folgenden Abschnitten wird beschrieben, wie das Standardformat (auch als System-Daten-Format SDF bezeichnet) erzeugt werden kann und welche Schritte unternommen werden müssen, um Dateien im Standardformat in ein Programm zu übernehmen.

```
Beispiel:
10 Byte Bezeichnung
 5 Byte  Ident-Nummer

interne Darstellung:
SCHRAUBE  00012<CR><LF>MUTTER    00013<CR><LF><SUB>
<erster Datensatz      ><zweiter Datensatz    >

Ausgabeformat:
SCHRAUBE  00012
MUTTER    00013
```

Bild 1 Aufbau einer Datei im Standardformat

2. BASIC-80

2.1. BASIC-80-Programme

In den meisten BASIC-Dialekten werden die Befehlswörter intern durch ein Kurzzeichen (Token) dargestellt, wodurch Speicherplatz und Ausführungszeit gespart werden. Eine Verarbeitung des Programms mittels Textprozessor wird dadurch ausgeschlossen. Soll das Programm in der lesbaren Form abgespeichert werden, so ist bei BASIC-80 die Option "A" anzugeben:
SAVE dateiname,A
Diese Option wird ebenfalls für das MERGE-Kommando und Compiler-Quelltexte verlangt. Durch die A-Option wird das Programm als ASCII-Datei auf Diskette abgelegt.

2.2. BASIC-Dateien

BASIC-80 arbeitet mit Direktzugriffs-(DA-) und sequentiellen Dateien. Beides sind reine ASACII-Dateien. Zahlen müssen vor der Ausgabe in Zeichenketten umgewandelt werden. Zur Einsparung von Speicherplatz ist die Konvertierung der Hexadezimaldarstellung in eine Zeichenkette mittels MKI$-, MKS$- oder MKD$-Befehl möglich, das soll hier aber nur am Rande erwähnt werden. In DA-Dateien können beliebige Datensätze jederzeit verändert werden, Lesen und Schreiben sind quasi gleichzeitig möglich, während sequentielle Dateien *nur zum Lesen* oder *nur zum Schreiben* eröffnet werden können. DA-Dateien eignen sich besonders für Einsatzfälle, in denen einzelne Datensätze einem häufigen Veränderungsprozeß unterliegen. Sie sind jedoch nicht zu anderen Darstellungen kompatibel. Nur sequentiell zum Schreiben eröffnete Dateien, deren Sätze mit
PRINT#n,variable;variable;...
geschrieben werden, erhalten das oben beschriebene Format.
Beispiel:
Sequentielle Datei:
Die folgenden Befehle erzeugen eine Datei, deren Aufbau Bild 1 entspricht.
OPEN"O",#2,dateiname
PRINT#2,variable;variable;variable
bzw. konkret
PRINT#2, "SCHRAUBE 00012"
Mit
CLOSE#2
wird die Datei geschlossen und das Dateiende SUB angefügt.
DA-Datei:
Die Vorgehensweise für die Erzeugung einer Direktzugriffsdatei ist eine andere:
OPEN"R",#1,dateiname,datensatzlaenge
FIELD#1,laenge AS puffervariable,...
RSET puffervariable=zeichenkettenvariable
PUT#1,datensatzzaehler
Der abgespeicherte Datenstrom für unser obiges Beispiel besitzt den Aufbau nach Bild 2. Es werden weder Datensatzende noch Datenende geschrieben, auf DA-Dateien ist die Funktion EOF nicht anwendbar.
Sollen DA-Dateien in anderen Programmsystemen verarbeitet werden, so müssen sie vorher in das sequentielle Format überführt werden (vgl. Bild 3). Die so erzeugte sequentielle Datei läßt sich mit Textprozessor, TURBO-PASCAL oder REDABAS weiterverarbeiten.

3. TURBO-PASCAL

TURBO-PASCAL verfügt über vielfältige Ausdrucksmittel für die Datenarbeit. Der we-

```
53 43 48 52 41 55 52 45 20 20 30 30 30 31 32
S  C  H  R  A  U  B  E        0  0  0  1  2

4D 55 54 54 45 52 20 20 20 20 30 30 30 31 33
M  U  T  T  E  R              0  0  0  1  3
```

Bild 2 Datenstrom einer DA-Datei (BASIC)

Bild 3 Umwandlung einer DA- in eine sequentielle Datei in BASIC

```
1000 OPEN "R",#1,dadatei,pufferlaenge : REM Eroeffnen DA-Datei
1010 FIELD #1,pufferlaenge AS puffervariable
1020 OPEN "O",#2,seqdatei :              REM Eroeffnen seq. Datei
1030 FOR I%=1 TO dateiende :             REM Lesen DA-Datei
1040    GET #1,I%
1050    PRINT #2,puffervariable :        REM Schreiben seq. Datei
1060 NEXT I%
1070 CLOSE :                             REM Schlieszen Dateien
```

Bild 4 Lesen einer sequentiellen Datei in TURBO-PASCAL

```
1       VAR      F:TEXT;              {Definition F als Textdatei}
2                S:STRING[15];        {Variable, auf die ein Daten-}
3                                     {feld eingelesen wird}
4                I:INTEGER;           {Zaehlvariable}
5
6       BEGIN
7          ASSIGN (F,dateiname);      {Zuweisung Dateiname-Filevar.}
8          RESET (F);                 {Eroeffnen fuer Lesen/Schreiben}
9          FOR I:=1 TO 2 DO           {Lesen 2 Datensaetze}
10            BEGIN
11            READLN (F,S);           {Lesen Textdatei}
12            WRITELN (S);            {Kontrolle auf Bildschirm}
13            END;
14         CLOSE (F);                 {Datei schlieszen}
15      END.
```

```
1       VAR      F:TEXT;
2                G:TEXT;
3                S:STRING[15];
4                I:INTEGER;
5
6       BEGIN
7          ASSIGN (F,dateiname);
8          RESET (F);
9          ASSIGN (G,dateiname);
10         REWRITE (G);
11         FOR I:=1 TO 3 DO           {Konvert. 3 Datensaetze}
12           BEGIN
13           READ (F,S);              {Lesen BASIC-DA-Datei}
14           WRITELN (G,S);           {Schreiben Turbo-Datei}
15           END;
16         CLOSE (F);
17         CLOSE (G);
18      END.
```

Bild 5 Umwandlung einer BASIC-DA-Datei in eine SDF-Datei mittels TURBO-PASCAL

```
(a)
Definition der REDABAS-Struktur:
Bezeichnung  BEZ     Zeichenkette C 10 Byte
Ident-Nummer IDENT   Zeichenkette C  5 Byte

S  C  H  R  A  U  B  E         0  0  0  1  2
53 43 48 52 41 55 52 45 20 20  30 30 30 31 32 0D 0A ...
==============================  ++++++++++++++
(Feld BEZ, 10 Byte)            (Feld IDENT, 5 Byte)

(b)
Definition der REDABAS-Struktur:
Bezeichnung  BEZ     Zeichenkette C  9 Byte
Ident-Nummer IDENT   Zeichenkette C  6 Byte

S  C  H  R  A  U  B  E         0  0  0  1  2
53 43 48 52 41 55 52 45 20  20 30 30 30 31 32 0D 0A ...
==========================  +++++++++++++++++
(Feld BEZ, 9 Byte)          (Feld IDENT, 6 Byte)
```

Bild 6 Zuordnung eines Datenstroms in REDABAS ▶

sentlichste Mangel des Standard-PASCAL, das Fehlen von Anweisungen für den Direktzugriff, wird mit dem Befehl

SEEK (dateivariable,datensatzzeiger)
für die Positionierung des Schreib-/Lesekopfes beseitigt.

Kompatibilität zu BASIC und anderen Programmsystemen ist mit TURBO-PASCAL-Dateien nicht gewährleistet. Als SDF-kompatibles Format ist der Dateityp TEXT anzusehen, auf den allerdings der Befehl SEEK nicht anwendbar ist. Mittels TEXT-Dateien werden maximal 255 Zeichen lange Datensätze eingelesen und weiterverarbeitet. Bild 4 zeigt ein Programm, das eine SDF-Datei in eine Zeichenkettenvariable mit 15 Byte Länge einliest und auf dem Bildschirm zur Kontrolle anzeigt. Mit der Anweisung READLN wird ein Datensatz bis CR LF gelesen. TURBO-PASCAL kann auch BASIC-80-Direktzugriffsdateien lesen, wenn in Zeile 11 mit der Anweisung READ gearbeitet wird. READ liest unabhängig von der Datensatzendekennung soviel ASCII-Zeichen, wie die Länge der Zuweisungsvariablen S angibt. Ebenso werden durch WRITE Datensätze ohne, durch WRITELN Datensätze mit CR LF als Ende erzeugt, sofern als Dateityp TEXT definiert wurde.
Im Unterschied zu BASIC-80 schließt TURBO-PASCAL auch mit WRITE geschriebene Dateien, deren Datensätze einen äquivalenten Aufbau wie DA-Dateien in BASIC besitzen, mit SUB ab. Das Beispielprogramm in Bild 5 wandelt eine BASIC-80-DA-Datei in eine TURBO-PASCAL-Datei im Standardformat um.

4. Textprozessor

Der Textprozessor TP arbeitet als einziges Programm durchgängig und ausschließlich mit dem Standardformat. Sinnvoll genutzt wird diese Möglichkeit zum Beispiel für die Verwendung von SDF-Kopien aus REDABAS und Diskettenprotokollen von KP. Darüber hinaus kann er Quelltexte für beliebige Programmiersprachen erzeugen. Es ist zu berücksichtigen, daß im Dokument-Modus Steuerzeichen eingefügt werden, die von anderen Programmsystemen eventuell nicht verarbeitet werden.

5. REDABAS

REDABAS-Dateien sind sequentiell organisiert. In einem Vorspann (Header) wird die Dateistruktur abgelegt. Dadurch ist die Verwendung von REDABAS-Dateien in anderen Programmsystemen nicht realisierbar. Mit der Option SDF in Verbindung mit COPY und APPEND werden Dateien im Standardformat erzeugt bzw. weiterverwendbar. Copy verhindert mit SDF die Erzeugung des Headers in der kopierten Datei, wobei die FOR- und die FIELD-Klausel uneingeschränkt wirksam bleiben:
COPY TO dateiname SDF
Ebenso bewirkt SDF als Ergänzung zu APPEND die Übernahme einer sequentiell organisierten Datei ohne Header:
APPEND FROM dateiname SDF
Entsprechend der definierten Struktur werden die Daten des eingehenden Stromes Byte für Byte den Feldern des REDABAS-Satzes zugeordnet. Mit dem Erkennen des Zeilenendes CR LF wird der nächste Datensatz aufgebaut. Bild 6 erläutert, wie die Zeichen eines Datenstroms bei unterschiedlicher Definition der Datenstruktur zugeordnet werden.

6. Kalkulationsprogramm

Das Kalkulationsprogramm erzeugt ASCII-Standarddateien im behandelten Format als Protokolldatei auf Diskette. Mit der Anweisung
/PB,bereich,D,dateiname
wird das Protokoll nicht auf Bildschirm oder Drucker, sondern auf Diskette erzeugt. Die Übernahme von KP-fremden Dateiinhalten ist trivial nicht möglich. KP ist damit das einzige Programmsystem, das keine Standarddateien weiterverarbeitet. Die Alternative ist die Umwandlung der ASCII-Datei in einen KP-Kommandostapel, in dem vor jedem Datensatz ein Sprungziel definiert wird. Das kann mit dem Textprozessor oder einem REDABAS-, BASIC- oder TURBO-PASCAL-Programm geschehen.
Beispiel:
Umwandlung einer Standarddatei in einen KP-Kommandostapel mittels Textprozessor:
Ursprungsdatei:
SCHRAUBE 00012
MUTTER 00013
Erzeugter Kommandostapel
=a5, "SCHRAUBE
=b5, "00012
=a6, "MUTTER
=b6, "00013

7. Schlußbemerkung

ASCII-Standarddateien bilden ein wertvolles Hilfsmittel für die Mehrfachnutzung von Dateien in unterschiedlichen Programmsystemen. Sie lassen sich mit BASIC, TURBO-PASCAL, REDABAS und KP erzeugen sowie in den drei erstgenannten übernehmen. Für Problemanalyse und Programmentwicklung sollten diese Möglichkeiten der Dateigestaltung konsequent genutzt werden.

UNIX-Gemeinschaft

Am 16.3.1988 fand im Leitzentrum für Anwendungsforschung Berlin die 1. Tagung der Entwickler- und Anwendergemeinschaft UNIX-kompatibler Software statt. Die Gemeinschaft stellt sich das Ziel, die Anwendung von UNIX-kompatiblen Betriebssystemen (MUTOS, PSU, WEGA, VMX u. a.) auf unterschiedlicher Hardware zu fördern und zur Erhöhung des Bestandes an Anwendungssoftware beizutragen. Interessenten für die Mitgliedschaft in dieser Gemeinschaft wenden sich an den
VEB Leitzentrum für Anwendungsforschung, Softwarebetrieb CAD/CAM, Jacques-Duclos-Str. 47/52, Berlin, 1156, Abt. F13, Tel. 37803480/481 (Koll. H. Grützbach).

Textverarbeitungssystem TEXT2 für KC 87

Ingo Rohner, Dresden

Ausgehend von den Erfahrungen bei der Entwicklung und beim Einsatz des Systems TEXT1 für KC 85/1 (Z9001) wurde für diese Gerätereihe ein neues Textsystem erstellt. Es ist wesentlich einfacher zu handhaben und trotzdem umfangreicher in seinen Möglichkeiten.

Einfacher Dialog

Der entscheidende Unterschied zwischen TEXT1 und TEXT2 besteht in der Nutzerführung. Der Nutzer muß nicht mehr aus einem Menü seine gewünschte Funktion auswählen, sondern kann sofort, wie bei einer Schreibmaschine, schreiben. Eine Unterscheidung zwischen Ersterfassung und Überarbeitung eines Textes gibt es nicht mehr. Ein zweiter wesentlicher Unterschied ist die Gestaltung der Schreibfläche. Da alle Rechner dieser Familie auf dem Bildschirm nur eine Darstellung von 40 Zeichen pro Zeile zulassen, mußte wiederum ein Kompromiß eingegangen werden. Dieser besteht aber nicht mehr in der Darstellung einer Textzeile über mehrere Bildschirmzeilen hinweg, sondern in der Nutzung des Bildschirms als „Lupe" über der Schreibfläche. Dies wird durch einen Rollmodus sowohl in horizontaler als auch in vertikaler Richtung erreicht. Zum Vereinfachen der Nutzerführung gehört die Fenstertechnik. In einem vom Nutzer definierbaren Fenster werden, abgehoben vom Text, alle Systemausschriften, Eingabeforderungen und Fehlermeldungen realisiert. Auch die Führung des Nutzers bei Unsicherheiten erfolgt über dieses Fenster.

Erweiterter Zeichensatz

Der Satz der verwendeten Zeichen wurde gegenüber dem Vorgängersystem wesentlich erweitert. Neu hinzugekommen sind die deutschen Umlaute ä, ö, ü, Ä, Ö und Ü sowie ß. Weiterhin stehen die eckigen und geschweiften Klammern zur Verfügung ([\ ] { : } ¯). Zu erreichen sind diese Zeichen über die LIST-Taste und eine Folgetaste. Die Umlaute können je nach Nutzerwunsch als inverse a, o, u... (bei Farbnutzung) oder als blinkende Zeichen (bei Nutzung eines Schwarzweißfernsehgerätes) dargestellt werden. Für die einfache Bestimmung von begrenzten Textmengen bei der Ausgabe oder der Gestaltung von Texten wurden Kennzeichen für den Blockbeginn und das Blockende eingeführt. Neu ist auch die Unterstützung von 8 verschiedenen Schriftarten und 8 verschiedenen Zeilenvorschüben. Vorgegeben werden nur die Maße des zu bedruckenden Papiers. Die Umrechnung der Schriftbreiten und der Zeilenvorschübe übernimmt TEXT2. Standard sind 10 CPI und 6 LPI. Wird nun z. B. eine Schriftart mit 12 CPI gewählt, so passen auf eine Standardzeile statt der sonst 64 möglichen Zeilen nun 76 Zeichen.

Gestaltung von Texten

Der Nutzer kann beliebig den linken Rand, die Zeilenlänge und die Seitengröße während des laufenden Betriebes neu einstellen. Für das Gestalten von Texten stehen folgende Kommandos zur Verfügung:

- Aufbereiten eines Abschnittes entsprechend den eingestellten Formaten
- Trennen der aktuellen Zeile an der Cursorposition in 2 Zeilen
- Verbinden der aktuellen Zeile mit der nachfolgenden
- Zentrieren von Zeilen
- Kopieren bzw. Transportieren von markierten Blöcken.

Weiterhin gibt es die Möglichkeit, Steuerzeichen für Unterstreichung, Fettdruck und Breitschrift in den Text einzufügen. Die automatische Gestaltung der Texte wird durch Schalter unterstützt, die zu jedem Zeitpunkt eingestellt werden können. Diese sind:

- Tastatur im Schreibmaschinen- oder Computermodus
- Insertmodus
- Wortumbruch am Zeilenende
- rechtsbündiges Ausrichten der Zeilen
- halbautomatische Worttrennung
- senkrechte Bewegung des Cursors.

Die Worttrennung wird durch einen Algorithmus realisiert, der mit einer Trefferquote von 94 % einen richtigen Trennvorschlag macht. Die senkrechte Bewegung des Cursors ist sinnvoll für die Erstellung von Grafiken und bedeutet, daß der Cursor bei Vertikalbewegungen nicht an das Ende der Zeile gestellt wird, falls sie kürzer als die Nachbarzeile ist, sondern exakt senkrecht bewegt wird.

Positionierung des Cursors

- Suchen eines Begriffes ohne Beachtung der Groß- und Kleinschreibung. Es gibt auch die Möglichkeit, von einem bestimmten Begriff bis zu einem anderen zu suchen.

Beispiel: Die Angabe von f#er findet die Begriffe Feuer, Füller und Fieber, jedoch auch das Wort fertig.

- Austauschen einer Zeichenkette gegen eine andere. Die einzusetzende Zeichenkette kann auch leer sein. Das Ersetzen kann mit oder ohne Sicherheitsabfrage erfolgen
- Stellen des Cursors an den Textanfang
- Stellen des Cursors an das Textende
- eine Seite zurück blättern
- eine Seite vorwärts blättern
- Positionieren auf den markierten Blockanfang
- Positionieren auf das markierte Blockende.

Ein- und Ausgabe

Die Eingabe von der Kassette wurde völlig erneuert. Für die Eingabe kann ein Filename angegeben werden, muß aber nicht. Es werden von TEXT2 selbständig die Aufzeichnungsformate von TEXT2, TEXT1, des BASIC-Interpreters (nur LIST#1-Dateien) und des IDAS (bzw. EDIT/ASM) erkannt und angepaßt. Geplant ist eine Erweiterung auf das Format des KC 85/2 (/3). Die Ausgabe von Texten ist auf Drucker und Kassette möglich. Die Menge des auszugebenden Textes kann der gesamte Text, der Text bestimmter Seiten oder der in Blockmarken eingeschlossene Text sein. Bei der Ausgabe auf Kassette kann in den Formaten von TEXT2, BASIC und IDAS aufgezeichnet werden. Geplant ist auch hier eine Erweiterung auf das Format des KC 85/3.

Gemeinsame Grundsprache für RGW-Software

Es ist ein gutes Jahr vergangen, seitdem wissenschaftliche Zentren der RGW-Länder mit gemeinsamen Arbeiten zum Problem „Entwicklung der Technologie der Ausarbeitung und industriellen Produktion von Software für Rechentechnik" begonnen haben. Daran beteiligen sich mehr als 70 Institutionen. Die gemeinsame Konzeption der Zusammenarbeit wurde eine Art Berufsmanifest, um das sich ehemals zersplitterte, einseitig ausgerichtete wissenschaftliche Einrichtungen konzentrieren. Ihre Hauptaufgabe besteht darin, einen Mechanismus von Wechselbeziehungen zu schaffen, der nach dem Prinzip „einen Teil produzieren – das Ganze schaffen" arbeitet. Dieses Prinzip entspricht wohl am besten den nationalen Interessen aller Partner: Die Programmierung ist heute einer der verantwortungsvollsten Zweige des wissenschaftlich-technischen Fortschritts, geht es doch um die weitere Vervollkommnung der Speicherfähigkeit – des Gehirns – moderner Computer. In erster Linie geht es um den schnellen Gewinn ressourcensparender Technologien sowie die Ausarbeitung und Produktion von Software in kürzester Zeit, die eine Steigerung der Arbeitsproduktivität um mindestens das Doppelte je Planjahrfünft gewährleisten würde. Natürlich bei drastischer Steigerung der Qualität und der Zuverlässigkeit der Software. Heute kann man sagen, daß bereits eine stabile Grundlage für die Zusammenarbeit in den kommenden Jahren geschaffen wurde. Das Institut für Kybernetik der Akademie der Wissenschaften der Ukrainischen SSR hat einen internationalen Wettbewerb für eine universelle Grundsprache für die Software der RGW-Länder durchgeführt. Insgesamt gingen 64 Arbeiten aus Bulgarien, Ungarn, der DDR, Polen, der Sowjetunion und der ČSSR ein. Mit diesem schöpferischen Wettbewerb konnte eine wertvolle „Ideenbank" zum Aufbau einer mobilen Technologie gesammelt werden. So wird hochqualitative Software entwickelt, die ohne Mühe bei unabhängiger Ausarbeitung in einzelnen Ländern miteinander gekoppelt werden kann. Dieser Wettbewerb war auch eine Art Messe von Lösungen vieler Probleme, doch der erste Platz wurde von niemandem erreicht, keine einzige Arbeit konnte voll und ganz eine Lösung des Problems beanspruchen, eine Grundsprache des RGW entwickelt zu haben. Jetzt arbeitet ein zeitweiliges internationales Expertenkollektiv daran. Wesentlichen wissenschaftlichen Nutzen bringen heute auch direkte Kooperationsbeziehungen, die auf Wirtschaftsverträgen beruhen. Ein Beispiel dafür liefert der Dresdner Betrieb „Robotron-Projekt", der mit dem bulgarischen Zentralinstitut für Rechentechnik und -technologie sowie der ungarischen Organisation für Rechentechnik „Samalk" zusammenarbeitet.

APN/ADN

Geplante Aktivitäten

Gegenwärtig ist ein Installationsprogramm in Arbeit, mit dem der Nutzer TEXT2 an seine speziellen Wünsche anpassen kann. Diese Anpassung betrifft die Gestaltung des Bildschirms, die Ansteuerung des Druckers und die Tastenfolgen zur Ausführung eines Kommandos. Die Ansteuerung des Druckers ist relativ unkompliziert, da zur Druckerausgabe nur die logische Systemschnittstelle genutzt wird (LD C,5 CALL 5). Alle druckerspezifischen Merkmale sind frei definierbar.

✉ KONTAKT ☏

Technische Universität Dresden
Informatikzentrum/Bereich AI, Koll. Rohner,
Mommsenstr. 13, Dresden, 8027

Tonbandinterface für universellen Datenaustausch

Stefan Bialluch, Hans-Friedrich Schaede

Oft besteht die Notwendigkeit, Daten zwischen Computern auszutauschen. Dabei ist es nicht immer möglich, die beiden Computer durch eine direkte Leitung zu verbinden (z. B. wegen örtlicher Trennung). In diesen Fällen ist es günstig, die Daten zunächst auf ein externes Speichermedium zu schreiben (Floppy-Disk, Kassette, EPROM) und dann in den anderen Computer einzulesen. Beide Computer müssen dazu das gleiche Interface besitzen. Für einen universellen Datenaustausch zwischen den verschiedenen Computern ist deshalb ein Interface notwendig, das sich einfach einbauen und anpassen läßt. Ein solches Interface soll hier in Einheit von Hardware und Software für alle U880-(Z80-)-Computer beschrieben werden.

Für die genannte Verwendung ergeben sich folgende Forderungen:

- *einfache Hardware*
- *variabler Anschluß des Computers*
- *einfache Implementierung der Software*
- *sichere Funktion mit verschiedenen (auch einfachen) Kassettenrecordern.*

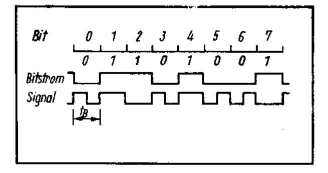


Bild 1 Kodierung: Diphase-Code

Eine Untersuchung bisher bekannter Interface-Systeme dieser Art hat ergeben, daß damit jeweils nur ein Teil dieser Forderungen erfüllbar ist /1/, /2/, /3/.

1. Kodierung und Format

Die Kodierung der Daten erfolgt wie schon in /1/, /2/, /3/ im Diphase-Code (Bild 1). Das heißt, an den Bitgrenzen wird grundsätzlich eine Flanke (Taktflanke) ausgegeben. Bei Null-Bits wird dann zwischen den Bitgrenzen eine zusätzliche Flanke ausgegeben.

Das Format (Bild 2) ist dem des Poly-Computers 880 /4/ sehr ähnlich. Zur Synchronisation werden 18 Null-Bits generiert. Danach folgt als Startbit eine Eins, anschließend ein festes Byte zur sicheren Synchronisation. Danach werden eine Dateinummer (1 Byte) und eine Blocknummer (2 Byte) ausgegeben. Es folgen ein Block von 32 Bytes Daten und die Prüfsumme (2 Byte).

Folgende Besonderheiten gibt es gegenüber dem PC-880-Format:

1. Beim PC-880 dient zur Synchronisation die Hälfte der sonst vorkommenden niedrigsten Frequenz. Untersuchungen mit verschiedenen Kassettenrecordern haben ergeben, daß teilweise Probleme bei der Übertragung dieser Frequenz auftreten. Dabei wurde die Bitwechselfrequenz relativ niedrig gewählt, um jederzeit einen universellen Datenaustausch und eine sichere Datenübertragung zu gewährleisten. Es traten unzulässige Verzerrungen bei der Synchronisationsfrequenz auf, so daß die nachfolgenden Daten nicht mehr gelesen werden konnten. Die vorgeschlagene Lösung kommt (auch zur Synchronisation) mit insgesamt 2 Frequenzen aus. Das Signal ist deshalb prinzipiell besser übertragbar (Bandbreite).
2. Zur Identifikation der einzelnen Blöcke wurde eine Datei- und eine Blocknummer in das Format aufgenommen. Die Dateinummer ist in allen Blöcken einer Datei gleich. Die Blocknummer wird jeweils inkrementiert, wobei am Anfang einer Datei stets mit 1 begonnen wird. Auf diese Weise ist es zum einen möglich, die gerade vorbeilaufende Datei beim Lesen vom Band zu identifizieren und somit auf die Bandposition zu schließen, zum anderen wird es möglich, gezielt ausgewählte Blöcke einzulesen, wenn nicht die komplette Datei benötigt wird. Außerdem wird verhindert, daß aufgrund eines Fehlers ein Block einfach ausgelassen wird. Das Leseprogramm erwartet stets nicht nur irgendeinen Block, sondern einen ganz bestimmten (Blocknummer).
3. Die Länge der Datei ist beim Lesen nicht immer vorab bekannt, Deshalb wird bei der Aufzeichnung am Ende der Datei ein Endekennzeichen gesendet. Dieses ist wie ein normaler Block aufgebaut, nur daß als Blocknummer ØFFFFH eingesetzt und danach abgebrochen wird (ohne Datenübertragung).

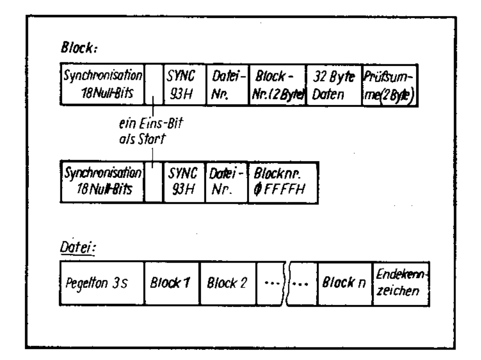


Bild 2 Format

2. Hardware

Die Hardware wurde so einfach wie möglich gehalten (Bild 3). Das Ausgangssignal wird nur durch einen Spannungsteiler gedämpft, um eine Übersteuerung des Recorders zu vermeiden. Zur Unterdrückung hochfrequenter Störungen ist der Kondensator C vorgesehen.

Das Wiedergabesignal muß zunächst verstärkt und dann auf einen Pegel von etwa 5 V begrenzt werden. Ein Trigger ist nur dann nötig, wenn der Eingang des Computers das erfordert (d. h. eine hohe Flankensteilheit, z. B. bei TTL).

Am Computer selbst können meist vorhandene Anschlüsse für das Interface als Ausgang und Eingang genutzt werden. Die einfachste Variante ist möglich, wenn zwei freie PIO-Pins (U 855) zur Verfügung stehen. Aber auch ein Pin reicht aus. Dieses wird dann als Ausgang oder Eingang abwechselnd genutzt. Sind keine PIO-Pins frei, sollte zunächst überprüft werden, ob andere Ein- und Ausgänge vorhanden sind, die sich eventuell zeitweise zweckentfremdet nutzen lassen. Bei TTL-Eingängen ist die Zwischenschaltung eines Triggers zu empfehlen (Bild 4).

Sind gar keine freien Ein-/Ausgänge vorhanden oder deren zweckentfremdete Nutzung nicht möglich, dann kann ein Ausgang durch ein D-Flipflop geschaffen werden. Als Eingang dient ein Gatter mit Tri-State-Ausgang (Bild 5).

Bild 4 Schaltungsbeispiel für Verstärker und Trigger (Verstärkerschaltung vom Polycomputer)

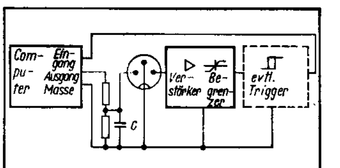


Bild 3 Blockschaltbild der Hardware

Bild 5 a) Ausgang mit D-Flip-Flop b) Eingang mit Tristate-Gatter

3. Software

Die Software wurde so gestaltet, daß sie sich auf verschiedenen Computern implementieren läßt.

Das Schreiben bzw. Lesen jeweils eines

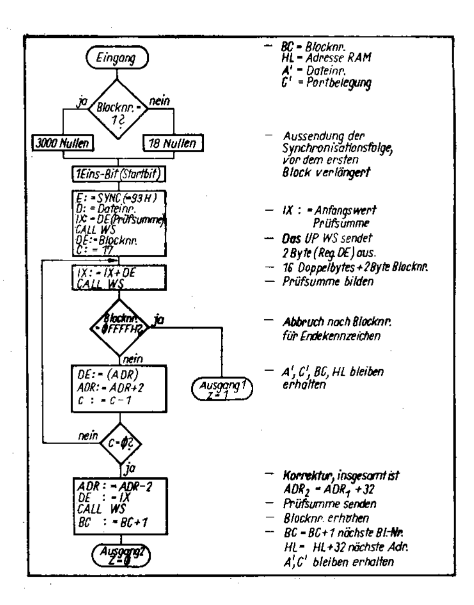


***Bild 6** Programmablaufplan für Unterprogramm BS*

***Bild 7** Programmablaufplan für Unterprogramm BL*

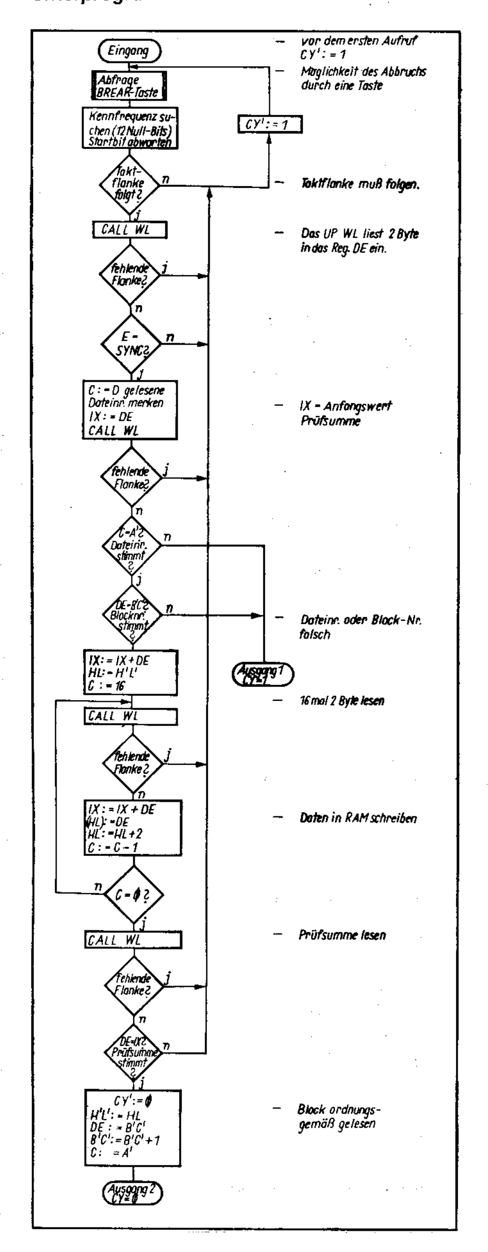


***Bild 8** Assembler-Listing der Unterprogramme BS und BL, außerdem ein Beispiel für das Unterprogramm BREAK* ▶

```
Kassetteninterface                    LISTING     PAGE 0001

ADDR   OBJECT    ST #

                 0002             GLOBAL  HP
                 0003             GLOBAL  HPDLG
>0093            0004 SYNC        EQU     93H
                 0005 ;NTA=750
>00F5            0006 PIO1BD      EQU     0F5H
>00F5            0007 OUTADR      EQU     PIO1BD
>00F5            0008 INADR       EQU     PIO1BD
>0002            0009 MASK        EQU     2
>0000            0010 BID         EQU     0
                 0011 ;
                 0012             ORG     2000H
'2000  C3FFFF    0013 IFUDA       JP      HP
                 0014 ;
                 0015 ;UP Block Schreiben
'2003  0B        0016 BS          DEC     BC
'2004  78        0017             LD      A,B
'2005  B1        0018             OR      C        ;Blocknr.=1 ?
'2006  03        0019             INC     BC
'2007  E5        0020             PUSH    HL
'2008  D9        0021             EXX
'2009  E1        0022             POP     HL       ;RAM-Adresse
'200A  112400    0023             LD      DE,36    ;18 Nullen
'200D  2003      0024             JR      NZ,BSM1-$         ;Bl.nr.<>1
'200F  11201C    0025             LD      DE,7200  ;Einpegelungsvorspann
'2012  0617      0026 BSM1        LD      B,BSN1   ;NTA/2
'2014  10FE      0027             DJNZ    0
'2016  CD5D20'   0028             CALL    FLOUT
'2019  1B        0029             DEC     DE
'201A  7A        0030             LD      A,D
'201B  B3        0031             OR      E
'201C  20F4      0032             JR      NZ,BSM1-$
'201E  0635      0033             LD      B,BSN2   ;NTA
'2020  10FE      0034             DJNZ    0        ;Zeit f.Startbit
'2022  CD5D20'   0035             CALL    FLOUT
'2025  08        0036             EX      AF,AF'
'2026  57        0037             LD      D,A      ;Dateinr.
'2027  08        0038             EX      AF,AF'
'2028  1E93      0039             LD      E,SYNC   ;Festwert
'202A  D5        0040             PUSH    DE
'202B  DDE1      0041             POP     IX       ;Anfangswert Pruefsumme
'202D  3E0D      0042             LD      A,BSN3   ;NTA/2
'202F  CD6420'   0043             CALL    WS       ;SYNC u.Dateinr.
'2032  D9        0044             EXX
'2033  C5        0045             PUSH    BC       ;Blocknr.
'2034  D9        0046             EXX
'2035  D1        0047             POP     DE
'2036  0611      0048             LD      B,17     ;Blocknr.+Dat.(32Byte)
'2038  3E0A      0049             LD      A,BSN4   ;NTA/2
'203A  DD19      0050 BSM3        ADD     IX,DE    ;Pruefsumme bilden
'203C  CD6420'   0051             CALL    WS
'203F  D9        0052             EXX
'2040  78        0053             LD      A,B      ;BC=0FFFFH?
'2041  A1        0054             AND     C
'2042  3C        0055             INC     A
'2043  C8        0056             RET     Z        ;ja:Ende Z=0
'2044  D9        0057             EXX
'2045  5E        0058             LD      E,(HL)   ;Daten aus RAM
'2046  23        0059             INC     HL
'2047  56        0060             LD      D,(HL)
'2048  23        0061             INC     HL
'2049  3E09      0062             LD      A,BSN5   ;NTA/2
'204B  10ED      0063             DJNZ    BSM3-$
'204D  2B        0064             DEC     HL
'204E  2B        0065             DEC     HL
'204F  DDE5      0066             PUSH    IX
'2051  D1        0067             POP     DE       ;Pruefsumme senden
'2052  3E07      0068             LD      A,BSN6   ;NTA/2
'2054  CD6420'   0069             CALL    WS
'2057  04        0070             INC     B        ;Z:=0
'2058  E5        0071             PUSH    HL
'2059  D9        0072             EXX
'205A  E1        0073             POP     HL
'205B  03        0074             INC     BC
'205C  C9        0075             RET
                 0076 ;BSN1=(NTA/2-81-11*WT)/(13+WT)
>0017            0077 BSN1        EQU     23
                 0078 ;BSN2=(NTA-76-11*WT)/(13+WT)
>0035            0079 BSN2        EQU     53
                 0080 ;BSN3=(NTA/2-162-22*WT)/(16+2*WT)
>000D            0081 BSN3        EQU     13
                 0082 ;BSN4=(NTA/2-207-27*WT)/(16+2*WT)
>000A            0083 BSN4        EQU     10
                 0084 ;BSN5=(NTA/2-235-33*WT)/(16+2*WT)
>0009            0085 BSN5        EQU     9
                 0086 ;BSN6=(NTA/2-259-37*WT)/(16+2*WT)
>0007            0087 BSN6        EQU     7
                 0088 ;
'205D  79        0089 FLOUT       LD      A,C      ;1 Flanke ausgeben
'205E  EE02      0090             XOR     MASK     ;Takte: 53+6*WT
'2060  4F        0091             LD      C,A
'2061  D3F5      0092             OUT     (OUTADR),A
'2063  C9        0093             RET
'2064  E5        0094 WS          PUSH    HL       ;2 Byte (DE) ausgeben
'2065  2E10      0095             LD      L,16     ;16 Bits
'2067  3D        0096 WSM1        DEC     A        ;E=111+14*WT
```

```
    Kassetteninterface                    LISTING     PAGE 0002

 ADDR  OBJECT    ST #

'2068  20FD      0097           JR     NZ,-1
'206A  CB3A      0098           SRL    D
'206C  CB1B      0099           RR     E       ;Bit ausschieben
'206E  3007      0100           JR     NC,WSM2-$
'2070  D5        0101           PUSH   DE      ;Zeitausgleich
'2071  D1        0102           POP    DE
'2072  D5        0103           PUSH   DE
'2073  D1        0104           POP    DE
'2074  00        0105           NOP
'2075  1803      0106           JR     WSM3-$
'2077  CD5D20'   0107 WSM2      CALL   FLOUT
'207A  3E14      0108 WSM3      LD     A,WSN1  ;NTA/2
'207C  3D        0109           DEC    A
'207D  20FD      0110           JR     NZ,-1
'207F  CD5D20'   0111           CALL   FLOUT
'2082  3E11      0112           LD     A,WSN2  ;NTA/2
'2084  2D        0113           DEC    L
'2085  20E0      0114           JR     NZ,WSM1-$
'2087  E1        0115           POP    HL
'2088  C9        0116           RET    ;A=38+5*WT
                 0117 ;
                 0118 ;WSN1=(NTA/2-54-7*WT)/(16+2*WT)
 >0014           0119 WSN1      EQU    20
                 0120 ;WSN2=(NTA/2-99-14*WT)/(16+2*WT)
 >0011           0121 WSN2      EQU    17
                 0122 ;UP Block Lesen
'2089  08        0123 BLA       EX     AF,AF'
'208A  37        0124           SCF    ;Fehler-Flag
'208B  08        0125           EX     AF,AF'
'208C  D9        0126 BL        EXX
'208D  E5        0127           PUSH   HL      ;RAM-Adr.
'208E  D9        0128           EXX
'208F  CD5521'   0129 BLX       CALL   BREAK   ;UP Abruchtaste abfr.
'2092  DBF5      0130           IN     A,(INADR)
'2094  47        0131           LD     B,A
'2095  2600      0132           LD     H,0
'2097  CD4121'   0133           CALL   FIFLA   ;Flanke suchen
'209A  28F3      0134           JR     Z,BLX-$
'209C  0E18      0135 BLM4      LD     C,24    ;12 Nullen
'209E  A8        0136 BLM1      XOR    B
'209F  47        0137           LD     B,A
'20A0  2605      0138           LD     H,BLN1  ;2*NTA/5
'20A2  CD4121'   0139           CALL   FIFLA
'20A5  20F5      0140           JR     NZ,BLM4-$
'20A7  2603      0141           LD     H,BLN2  ;NTA/5
'20A9  CD4121'   0142           CALL   FIFLA
'20AC  28E1      0143           JR     Z,BLX-$
'20AE  0D        0144           DEC    C
'20AF  20ED      0145           JR     NZ,BLM1-$
'20B1  A8        0146 BLM2      XOR    B       ;12 Nullen gefunden
'20B2  47        0147           LD     B,A
'20B3  260C      0148           LD     H,BLN3  ;3*NTA/4
'20B5  CD4121'   0149           CALL   FIFLA   ;Startbit suchen
'20B8  20F7      0150           JR     NZ,BLM2-$
'20BA  2608      0151           LD     H,BLN4  ;NTA/2
'20BC  CD4121'   0152           CALL   FIFLA   ;Taktfl.
'20BF  2005      0153           JR     NZ,BLM3-$
'20C1  08        0154 BLM6      EX     AF,AF'  ;Fehler
'20C2  37        0155           SCF            ;Fehler-Flag:=1
'20C3  08        0156           EX     AF,AF'
'20C4  18C9      0157           JR     BLX-$
'20C6  3E1C      0158 BLM3      LD     A,BLN5  ;3*NTA/4
'20C8  CD2521'   0159           CALL   WL      ;E=SYNC
'20CB  28F4      0160           JR     Z,BLM6-$;D=Dateinr.
'20CD  7B        0161           LD     A,E
'20CE  FE93      0162           CP     SYNC
'20D0  20EF      0163           JR     NZ,BLM6-$
'20D2  4A        0164           LD     C,D     ;Synchronis.i.O.
'20D3  D5        0165           PUSH   DE
'20D4  DDE1      0166           POP    IX      ;Anfangsw.Pruefs.
'20D6  3E17      0167           LD     A,BLN6
'20D8  CD2521'   0168           CALL   WL      ;Blocknr.
'20DB  28E4      0169           JR     Z,BLM6-$
'20DD  08        0170           EX     AF,AF'
'20DE  67        0171           LD     H,A     ;ges.Dateinr.
'20DF  08        0172           EX     AF,AF'
'20E0  79        0173           LD     A,C     ;gef. D.nr.
'20E1  BC        0174           CP     H
'20E2  2803      0175           JR     Z,BLM7-$
'20E4  37        0176 BLM5      SCF    ;Fehler-Return
'20E5  E1        0177           POP    HL
'20E6  C9        0178           RET
'20E7  D9        0180 BLM7      EXX
'20E8  C5        0181           PUSH   BC      ;ges.Blocknr.
'20E9  D9        0182           EXX
'20EA  E1        0183           POP    HL
'20EB  ED52      0184           SBC    HL,DE   ;Vergleich
'20ED  20F5      0185           JR     NZ,BLM5-$
'20EF  DD19      0186           ADD    IX,DE
'20F1  0E10      0187           LD     C,16    ;16*2 Bytes
'20F3  3E14      0188           LD     A,BLN7  ;3*NTA/4
'20F5  CD2521'   0189 BLM8      CALL   WL
'20F8  E1        0190           POP    HL
'20F9  288E      0191           JR     Z,BLA-$
'20FB  DD19      0192           ADD    IX,DE
```

Blocks ist eine relativ autonome Sache, da jeder Block seine eigene Synchronistation hat. Die Software basiert daher auf zwei Unterprogrammen (Bilder 6 und 7):

1. UP Block Schreiben (BS)
2. UP Block Lesen (BL).

Diese Unterprogramme haben definierte Ein- und Austrittsbedingungen, die eine einfache Einbindung in ein entsprechendes Hauptprogramm ermöglichen. Sie benutzen keine RAM-Arbeitszellen und sind frei von jeglichem Bediendialog. Die Programme arbeiten ohne Interrupt. Das trägt zu einer variablen Gestaltung der Hardware bei. Die Zeiten werden durch Programmzeitschleifen gebildet. Deshalb ist während des Programmlaufes auch kein anderer Interrupt zulässig. Aus diesem Grund läßt sich ohne Bedenken der Zweitregistersatz des U 880 verwenden. Die Zeitkonstanten für die Zeitschleifen ergeben sich aus der Taktfrequenz des Computers, der gewünschten Zeit und der Zeit für andere dazwischenliegende Befehle. Sie lassen sich nach den angegebenen Formeln (siehe Bild 8) leicht berechnen. Die Variable NTA ist die Anzahl der Takte pro Bitzeit. Sie ergibt sich zu

$$NTA = \frac{f_T}{f_{BW}} \text{ oder } NTA = f_T \cdot t_B$$

f_T = Taktfrequenz des Computers
f_{BW} = Bitwechselfrequenz
t_B = Bitzeit

In manchen Computern wird der **M1-Zyklus** durch **WAIT**-Takte verlängert. Diese werden durch die Variable WT berücksichtigt (WT = Anzahl der WAIT-Takte im M1-Zyklus). Ohne zusätzliche WAIT-Takte ist WT = 0. Bei kleineren Werten von NTA (< 600) oder höheren Werten von WT (> 5) werden einige Zeitkonstanten zu klein (≤ 3) und damit der Rundungsfehler zu hoch. In diesem Fall ist die Funktion unsicher.

Moderne Assembler ermöglichen die automatische Berechnung der Zeitkonstanten. Dazu sind die Formeln direkt im Quelltext einzutragen (EQU).

Für Nachnutzungen wird an dieser Stelle empfohlen, die Bitwechselfrequenz im Interesse eines universellen Datenaustausches auf einheitlich 1200 Hz festzulegen (entspricht einer Bitzeit von t_B = 0,833 ms). Diese relativ niedrige Frequenz und damit Datenübertragungsrate ist orientiert an der Einsetzbarkeit auch schlechter Kassettenrecorder und an einer sicheren Datenübertragung. Auch die Verwendung verschiedener Kassettenrecorder für Aufnahme und Wiedergabe bringt oft einen Qualitätsabfall mit sich.

3.1. Unterprogramm Blockschreiben (BS)

Das UP BS (Bild 6) realisiert das oben angegebene Format. Vor dem 1. Block der Datei wird die Synchronisationsfolge zur Einpegelung des Recorders verlängert (etwa 3 s).
Folgende Werte sind in den Registern zu übergeben:

BC = Blocknummer (1 = 1. Block)
HL = Adresse im RAM, Quelle der Daten
A' = Dateinummer
C' = Belegung des Ausgabeports.

Da die Ausgabebefehle des U 880 sich jeweils auf ein ganzes Byte beziehen, muß im Register C' eventuell die Belegung des benutzten Ausgabeports übergeben werden (bei PIO: IN A, (n); LD C,A; EXX). Vom UP BS wird nur das festgelegte Bit verändert.

Ist die Blocknummer (BC) = 0FFFFH, wird ein Endekennzeichen gesendet. Dabei bleiben die ausgegebenen Registerinhalte erhalten (Z = 1).
Ist ein Block gesendet worden, gilt bei der Rückkehr folgende Registerbelegung (Z = 0):
BC = BC + 1 (nächste Blocknummer)
HL = HL + 32 (nächste Adresse im RAM)
A' und C' bleiben erhalten.
Folgende Register werden zerstört: AF, B'C' D'E', H'L', IX.
Zwischen den UP-Aufrufen (Call BS) ist im Hauptprogramm (Bild 9) Gelegenheit für Dialogarbeit (Ausgabe von Blocknummern und Adresse). Die dadurch entstehende Pause kann theoretisch beliebig lang sein. Sie sollte jedoch im Interesse einer hohen Datenübertragungsrate klein gehalten werden.

3.2. Unterprogramm Blocklesen (BL)

Das UP BL (Bild 7) sucht zuerst eine Synchronisationsfolge (mindestens 12 Nullbits). Danach liest es die Dateinummer und die Blocknummer. Anschließend werden beide mit den Sollwerten verglichen. Wenn sie übereinstimmen, werden die nachfolgenden Daten gelesen und in den RAM (ab Adresse HL) geschrieben. Fehlen Taktflanken oder stimmt die Prüfsumme nicht, wird der Einlesevorgang abgebrochen. Dabei erfolgt keine Rückkehr zum Hauptprogramm, sondern ein Neustart des UP BL. Außerdem wird das Flag CY' = 1 gesetzt.
Ist ein Block ohne Fehler eingelesen worden, wird das Flag zurückgesetzt (CY' = 0) und mit CY = 0 zum Hauptprogramm zurückgekehrt. Stimmt die Datei- oder Blocknummer nicht mit der gesuchten überein, erfolgt sofort ein RET zum Hauptprogramm mit CY = 1.
Bis zur nächsten Synchronisationsfolge bleibt eine längere Zeit für Dialogarbeit (Zeit max. 272 × NTA). Diese reicht sogar, um eine Multiplexanzeige bei einfachen Computern kurzzeitig einzuschalten. Dabei kann die vorgefundene Datei- und Blocknummer angezeigt werden. Dadurch ist jederzeit eine Orientierung auf der Kassette möglich.
Der Fall einer falschen Blocknummer tritt auch ein, wenn ein Endekennzeichen gelesen wird. Die gelesene Blocknummer ist daraufhin auf 0FFFFH zu überprüfen (Bild 10). Des weiteren muß geprüft werden, ob die Datei ordnungsgemäß eingelesen wurde. Das geschieht anhand des Flags CY'. Ist es rückgesetzt, kann der Einlesevorgang ordnungsgemäß beendet werden. Ist es aber gesetzt, muß das Endekennzeichen ignoriert werden. Deshalb ist dieses Flag im Hauptprogramm vor dem ersten Aufruf von BL zu setzen (CY' = 1).
Ansonsten gilt für den Aufruf folgende Registerbelegung:
A' = Dateinummer
B'C' = Blocknummer
H'L' = Adresse RAM, Ziel der Daten.
Das UP hat wie oben beschrieben 2 Ausgänge (RET):
1. CY = 1 Datei- oder Blocknummer ist falsch

C	gelesene Dateinummer	
DE	gelesene Blocknummer	
A'	gesuchte Dateinummer	
B'C'	gesuchte Blocknummer	} bleibt
H'L'	Adresse RAM	} erhalten

2. CY = 0 Block ist richtig eingelesen
CY' = 0
B'C' = B'C' + 1 (nächste Blocknummer)
H'L' = H'L' + 32 (nächste RAM-Adresse)
C = A' (Dateinummer)
DE = B'C' − 1 (gelesene Blocknummer).
Im praktischen Betrieb kommt es vor, daß ein Block auch nach mehrmaligen Lesevorgängen nicht mehr gelesen werden kann (z. B. Fehler im Bandmaterial). In diesem Fall muß das Leseprogramm (BL) abgebrochen werden. Zu diesem Zweck ist ein UP-Ruf vorgesehen (Call BREAK). Dieses Programm ist vom Anwender zu erstellen, da es sich um eine Tastaturabfrage handelt, welche stark von den Gegebenheiten am jeweiligen Computer abhängt.
Dazu folgende Hinweise (s. a. Beispiel Bild 8, Listing ab Zeile 280):
– Kurze Laufzeit (zur Orientierung: max. 2 × NTA).
– Die Register A'F', B'C', H'L' dürfen nicht verändert werden.
– Ist die entsprechende Taste nicht gedrückt, dann RET.
– Ist die Taste gedrückt: Entkellerung der RET-Adresse zu UP BL, Entkellerung Adresse HL und der RET-Adresse zum Hauptprogramm. Danach Sprung zur Auswertung im Hauptprogramm (Dialog).
– Das UP kann auch entfallen (evtl. durch NOP ersetzen).

4. Hinweise zur Implementierung

Das Assembler-Listing für die Unterprogramme BS und BL ist im Bild 8 angegeben. Assembliert wurde für den Lerncomputer LC-80 /5/. Für diesen ist daher sofort der Maschinencode nutzbar (erst sinnvoll bei RAM-Erweiterung). Das Interface läuft mit dem vorhandenen Diodenanschluß. Die Taktfrequenz beim LC-80 beträgt etwa 900 kHz. Dafür und für f_B = 1 200 Hz sind die Zeitkonstanten berechnet. Folgende Zeilen sind zur Nachnutzung auf anderen Computern entsprechend zu ändern:
– Datenbit für Eingang, Zeile 10
– Maske für Ausgang, Zeile 9
– Zeitkonstanten
BS, Zeilen 76–87, 118–121
BL, Zeilen 224–239, 256–259
– Adressenbezug, Zeile 12
– I/O-Adressen
BS, Zeile 7
BL, Zeile 8

4.1. Hauptprogramm

Das Hauptprogramm ist entsprechend den Gegebenheiten des jeweiligen Computers vom Anwender selbst zu erstellen.
Am Beginn des Interfaceprogrammes stehen selbstverständlich zunächst der Eingabedialog für die Variablen und evtl. eine Initialisierung der I/O-Bausteine (PIO). Die nachfolgende Aufrufschleife erfordert eine besondere Beachtung der Zeitbedingungen. Dieser Teil ist in den Programmablaufplänen (Bil-

```
      Kassetteninterface                    LISTING      PAGE 0003

 ADDR   OBJECT     ST #

'20FD   73         0193          LD      (HL),E   ;Daten in RAM
'20FE   23         0194          INC     HL
'20FF   72         0195          LD      (HL),D
'2100   23         0196          INC     HL
'2101   E5         0197          PUSH    HL
'2102   3E15       0198          LD      A,BLN8   ;3*NTA/4
'2104   0D         0199          DEC     C
'2105   20EE       0200          JR      NZ,BLM8-$
'2107   CD2521'    0201          CALL    WL       ;Pruefsumme lesen
'210A   E1         0202          POP     HL
'210B   CA8920'    0203          JP      Z,BLA
'210E   EB         0204          EX      DE,HL
'210F   DDE5       0205          PUSH    IX
'2111   C1         0206          POP     BC
'2112   A7         0207          AND     A
'2113   ED42       0208          SBC     HL,BC    ;Test Pruefs.
'2115   EB         0209          EX      DE,HL
'2116   C28920'    0210          JP      NZ,BLA
'2119   08         0211          EX      AF,AF'
'211A   4F         0212          LD      C,A      ;Dateinr.
'211B   A7         0213          AND     A        ;Fehler-Flag:=0
'211C   08         0214          EX      AF,AF'
'211D   E5         0215          PUSH    HL       ;neue Adr.
'211E   D9         0216          EXX
'211F   E1         0217          POP     HL       ;uebernehmen
'2120   C5         0218          PUSH    BC       ;Blocknr.
'2121   03         0219          INC     BC       ;+1
'2122   D9         0220          EXX
'2123   D1         0221          POP     DE       ;alte Bl.nr.
'2124   C9         0222          RET      ;CY=0, Z=1
                   0223 ;
                   0224 ;BLN1=(2*NTA/5-67-8.5*WT)/44+7*WT
 >0005             0225 BLN1     EQU     5
                   0226 ;BLN2=(NTA/5-36-4*WT)/(44+7*WT)
 >0003             0227 BLN2     EQU     3
                   0228 ;BLN3=(3*NTA/4-56-6.5*WT)/(44+7*WT)
 >000C             0229 BLN3     EQU     12
                   0230 ;BLN4=(NTA/2-36-4*WT)/(44+7*WT)
 >0008             0231 BLN4     EQU     8
                   0232 ;BLN5=(3*NTA/4-119-15.5*WT)/(16+2*WT)
 >001C             0233 BLN5     EQU     28
                   0234 ;BLN6=(3*NTA/4-190-26.5*WT)/(16+2*WT)
 >0017             0235 BLN6     EQU     23
                   0236 ;BLN7=(3*NTA/4-248-35.5*WT)/(16+2*WT)
 >0014             0237 BLN7     EQU     20
                   0238 ;BLN8=(3*NTA/4-221-29.5*WT)/(16+2*WT)
 >0015             0239 BLN8     EQU     21
'2125   2E10       0240 WL       LD      L,16     ;2 Byte lesen ->DE
'2127   3D         0241 WLM2     DEC     A
'2128   20FD       0242          JR      NZ,-1
'212A   CD4B21'    0243          CALL    BITIN
'212D   2801       0244          JR      Z,WLM1-$
'212F   37         0245          SCF      ;bei Z=0: CY:=1
```

```
      Kassetteninterface                   LISTING      PAGE 0004
 ADDR  OBJECT     ST #

 '2130  CB1A      0246 WLM1     RR      D           ;einschieben
 '2132  CB1B      0247          RR      E
 '2134  2607      0248          LD      H,WLN1   ;NTA/2
 '2136  CD4121'   0249          CALL    FIFLA
 '2139  C8        0250          RET     Z           ;Fehler Z=1
 '213A  3E1D      0251          LD      A,WLN2   ;3*NTA/4
 '213C  2D        0252          DEC     L
 '213D  20E8      0253          JR      NZ,WLM2-$
 '213F  2C        0254          INC     L           ;Z:=0
 '2140  C9        0255          RET
                  0256 ;WLN1=(NTA/2-69.5-9.5*WT)/(44+7*WT)
  >0007           0257 WLN1     EQU     7
                  0258 ;WLN2=(3*NTA/4-96-13.5*WT)/(16+2*WT)
  >001D           0259 WLN2     EQU     29
                  0260 ;
                  0261 ;UP Flanke abwarten, zeitbegrenzt
 '2141  DBF5      0262 FIFLA    IN      A,(INADR)
 '2143  A8        0263          XOR     B
 '2144  CB47      0264          BIT     BID,A
 '2146  C0        0265          RET     NZ          ;Z=0, Flanke gef.
 '2147  25        0266          DEC     H
 '2148  20F7      0267          JR      NZ,FIFLA-$
 '214A  C9        0268          RET                 ;Z=1, keine Fl.
                  0269 ;
                  0270 ;UP Bit abfragen
 '214B  DBF5      0271 BITIN    IN      A,(INADR)
 '214D  A8        0272          XOR     B           ;Vergleich
 '214E  CB47      0273          BIT     BID,A
 '2150  F5        0274          PUSH    AF
 '2151  A8        0275          XOR     B           ;Uebernahme
 '2152  47        0276          LD      B,A         ;d.neuen Wertes
 '2153  F1        0277          POP     AF          ;CY=0, Z=1: gleich
 '2154  C9        0278          RET                 ;       =0: ungleich
                  0279 ;
                  0280 ;Beisiel: Abbruch-UP (BREAK fuer LC80)
  >00F9           0281 PIO2BD   EQU     0F9H
 '2155  3EF8      0282 BREAK    LD      A,0F8H
 '2157  D3F5      0283          OUT     (PIO1BD),A
 '2159  DBF9      0284          IN      A,(PIO2BD)
 '215B  CB7F      0285          BIT     7,A
 '215D  3EFC      0286          LD      A,0FCH
 '215F  D3F5      0287          OUT     (PIO1BD),A
 '2161  C0        0288          RET     NZ          ;Taste nicht bet.
 '2162  E1        0289          POP     HL          ;RET-Adr.zu BL
 '2163  E1        0290          POP     HL          ;Kellerung HL
 '2164  E1        0291          POP     HL          ;RET-Adr.zum HP
 '2165  C3FFFF    0292          JP      HPDLG       ;Dialog im HP
                  0293 ;
                  0294          END     IFUDA

 ERRORS=0000
```

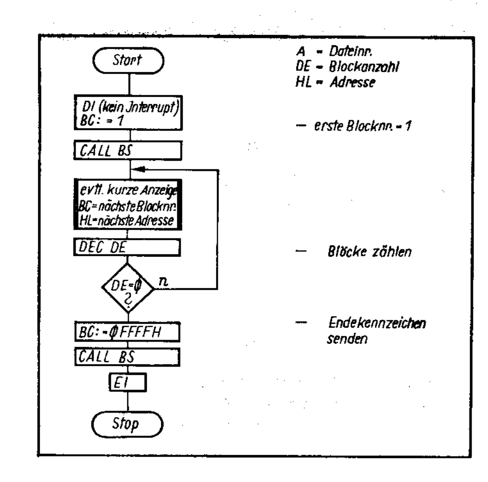


Bild 9 Programmablaufplan für Datei Schreiben

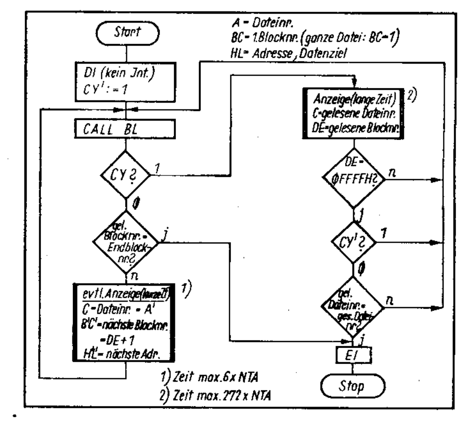


Bild 10 Programmablaufplan für Datei Lesen (auch teilweise)

der 9, 10) dargestellt. Die verschiedenen Computer lassen sich dazu in zwei Kategorien einteilen:

1. Einfache Lerncomputer (z.B. LC-80, Polycomputer 880, Prozeßsteuersysteme)

Bei diesen Computern wird eine LED-Anzeige zusammen mit einer Hexadezimaltastatur multiplex betrieben. Da während des Laufes der Interfaceprogramme die volle Prozessorzeit gebunden ist, kann die Anzeige nicht bedient werden. Um dennoch die vorbeilaufenden falschen Blöcke beim Lesen vom Band anzeigen zu können, wurde im UP BL ein Austritt (RET) nach dem Lesen der Blocknummer vorgesehen. Die Zeit, in der dann der Datenblock vorbeiläuft, ist deshalb zur Betätigung der Multiplexanzeige nutzbar. In dem entsprechenden Programmteil ist die Laufzeit selbst zu begrenzen (auf max. 272 × NTA). Wird die Zeit nicht eingehalten, kann die Synchronisation des nachfolgenden Blockes verpaßt werden.

An den anderen Ausgabepunkten (zwischen den UP-Rufen BS, BL) ist die Bedienung der Multiplexanzeige zeitlich nicht möglich. Zur Kontrolle des Schreib- bzw. Lesevorgangs kann die Anzeige, wenn überhaupt, nur noch direkt genutzt werden (einschalten einer einzelnen Stelle). Das gleiche gilt für die Tastaturabfrage im UP BREAK. Eine normale Multiplexabfrage ist an dieser Stelle nicht möglich.

2. Kleincomputer, Home-Computer, Personalcomputer

Diese Computer besitzen zur Ausgabe einen Bildschirm und zur Eingabe eine alphanumerische Tastatur. Die Ausgabe auf den Bildschirm erfordert nur wenig Zeit, ist also an allen genannten Stellen möglich. Eine gewisse Sonderstellung nehmen in dieser Einteilung Computer ein, bei denen der Prozessor mit am Bildaufbau beteiligt ist (MC 80, ZX 81). Auf diese soll hier nicht näher eingegangen werden. Eine Implementierung dürfte jedoch grundsätzlich möglich sein.

5. Schlußbemerkung

Der universelle Austausch von Daten und Programmen ist oft von großem Vorteil. Deshalb sollte jeder Computer über entsprechende Kommunikationsmittel verfügen. Das beschriebene Kassetteninterfacesystem hat den Vorteil, daß es sich leicht an verschiedene Computer – vom einfachen Lerncomputer bis zum Bürocomputer und eventuell auch Prozeßrechner – anpassen läßt. In diesem Zusammenhang sei noch einmal auf die empfohlenen Bitrate von 600 Bit/s (Bitwechselfrequenz = 1 200 Hz) hingewiesen, deren Einheitlichkeit ja auch Voraussetzung für einen Datenaustausch ist.

Besitzern von Computern mit anderen Prozessoren mag dieser Beitrag als Anregung dienen, auf ihrem Computer ein Interface mit gleichen äußeren Eigenschaften (Code, Format, Bitrate) zu realisieren und eventuell auch zu publizieren.

Literatur

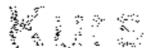
/1/ Weisse, D.: Magnetbandspeicher nach dem Diphase-Verfahren. Radio, Ferns., Elektron., Berlin 25 (1976) 24, S. 789–791

/2/ Troll, A.; Hübner, U.: Daten- und Programmabspeicherung auf Heimmagnetbandgeräten. Radio, Ferns., Elektron., Berlin 31 (1982) 12, S. 796–799

/3/ Will, B.: Schnelles Tonbandinterface für den U 880. Radio, Ferns., Elektron., Berlin 34 (1985) 10, S. 667–669

/4/ Burkhard, S.; Hübner, N.: Technik und Anwendung des Poly-Computers 880. Radio, Ferns., Elektron., Berlin 33 (1984) 5, S. 282–287

/5/ Kämpf, W.: Lerncomputer LC-80. Radio, Ferns., Elektron., Berlin 33 (1984) 10, S. 669–672



Mikroprozessorsystem K 1810 WM86

Hardware · Software · Applikation (Teil 2)

Prof. Dr. Bernd-Georg Münzer
(wissenschaftliche Leitung),
Dr. Günter Jorke, Eckhard Engemann,
Wolfgang Kabatzke, Frank Kamrad,
Helfried Schumacher, Tomasz Stachowiak
Wilhelm-Pieck-Universität Rostock,
Sektion Technische Elektronik,
Wissenschaftsbereich Mikrorechentechnik/
Schaltungstechnik

3. Interface-Schaltkreise

Im 16-Bit-Mikroprozessorsystem 8086 werden folgende programmierbare Interface-Schaltkreise als parallele, serielle und Timer-Ports verwendet:

8253 Programmierbarer Zähler/Zeitgeberschaltkreis, PIT
8251A Programmierbarer serieller Interfaceschaltkreis, USART
8255A Programmierbarer paralleler Interfaceschaltkreis, PPI

Diese Schaltkreise wurden nicht speziell für das 16-Bit-System 8086 entwickelt, sondern aus dem 8-Bit-System 8080 übernommen und bezüglich der dynamischen Parameter weiterentwickelt. Bei den Schaltkreisen USART und PPI sind in 16-Bit-Systemen nur die A-Typen anzuwenden.

3.1. Programmierbarer Zähler-/Zeitgeberschaltkreis 8253 (PIT)

Der PIT-Schaltkreis (Programmable Interval Timer) realisiert im Mikroprozessorsystem 8086 die Zähl- und Zeitgeberfunktionen und weist folgende wesentliche Leistungsmerkmale auf:

- 3 unabhängige 16-Bit-Zähler mit Zähleingang, Zählausgang und Gatesteuerung
- 6 programmierbare Betriebsarten
- Zählen im Binär- oder BCD-Format
- max. Zählfrequenz von 2,0 MHz.

3.1.1. Architektur

Die Funktionseinheiten des PIT (Bild 3.1) haben folgende Aufgaben:

● **Datenbuspuffer**
Der bidirektionale/tristate 8-Bit-Puffer stellt die Schnittstelle zum Systemdatenbus dar. Das Senden oder Empfangen von Daten erfolgt mit E/A-Operationen der CPU. Über den Datenbuspuffer werden drei Basisfunktionen des PIT ausgeführt:
- Programmieren der Modes
- Laden des Zählregisters
- Lesen des Zählerstandes.

● **Lese-/Schreib-Logik**
Die Lese-/Schreib-Logik verarbeitet die Informationen der Eingänge Read $\overline{RD}$, Write $\overline{WR}$, Chip Select $\overline{CS}$ sowie die Portadressen A0 und A1 des Systemsteuerbusses. Die Dekodierung für die Portadressen und deren Zuordnung zu den internen Funktionseinheiten ist in Tafel 3.1 dargestellt.

Tafel 3.1 *8253 Portadressen*

A1	A0	Port
0	0	Zähler 0
0	1	Zähler 1
1	0	Zähler 2
1	1	Control (nur Schreiben)

● **Steuerwortregister**
Das Steuerwortregister (A0 = A1 = 1) verarbeitet die Information des Datenbuspuffers als Steuerwort zur Grundinitialisierung des PIT. Das Steuerwortregister kann nur beschrieben werden. Über OUT-Befehle der CPU zum Steuerwortregister erfolgt die unabhängige Modeeinstellung jedes Zählers.

● **Zähler 0–2**
Der Aufbau aller drei Zähler ist identisch. Jeder Zähler besteht aus einem voreinstellbaren 16-Bit-Rückwärtszähler ohne Vorteiler zum Zähltakt CLK. Das Zählen kann im Binär- oder BCD-Format erfolgen. Zähler-Eingang CLK, Toreingang GATE und Ausgang OUT sind entsprechend der Modeselektion verknüpft. Mit dem Toreingang GATE können für jeden Kanal die Zählvorgänge getriggert, gestartet und gestoppt werden. Alle Zähler arbeiten unabhängig und beeinflussen sich gegenseitig nicht. Auch die Modeeinstellung und das Laden von Zählwerten erfolgt für jeden Zähler getrennt. Das Lesen der aktuellen Zählerinhalte wird durch CPU-IN-Operationen auch während des Lesevorgangs realisiert.

3.1.2. Pinbelegung

Der PIT 8253 wird in einem 24-poligen Standard-DIL-Gehäuse gefertigt (Bild 3.2). Die Anschlüsse haben folgende ***Funktionen:***

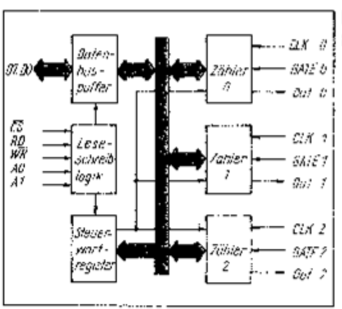


Bild 3.1 Architektur 8253-PIT

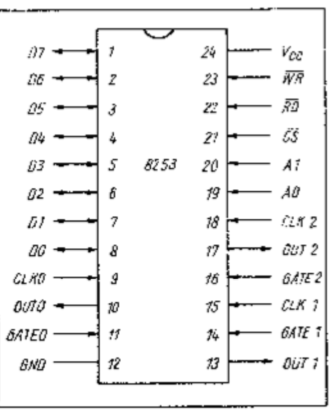


Bild 3.2 Anschlußbelegung 8253-PIT

D7–D0	8-Bit-Datenbus, bidirektional, 3-state
CLK 0–2	Zähler-Takteingänge
GATE 0–2	Toreingänge
OUT 0–2	Zählerausgänge
$\overline{\textbf{RD}}$	Read Lesesignal, Eingang low-aktiv
$\overline{\textbf{WR}}$	Write Schreibsignal, Eingang low-aktiv
$\overline{\textbf{CS}}$	Chip Select Steuersignal zur Bausteinauswahl, Eingang low-aktiv Der Anschluß wird über einen Dekoder 8205 zum System-Adreßbus geführt.
A0, A1	Adreßeingänge für die Auswahl der Zähler 0, 1, 2 bzw. Steuerwortregister (Tafel 3.1). Diese Anschlüsse sind mit den Adressen AB1 und AB2 des Systems zu verbinden.
Vcc	Betriebsspannung + 5 V
GND	Masseanschluß

3.1.3. Betriebsarten

Die Zähler 0, 1, 2 des 8253 werden individuell

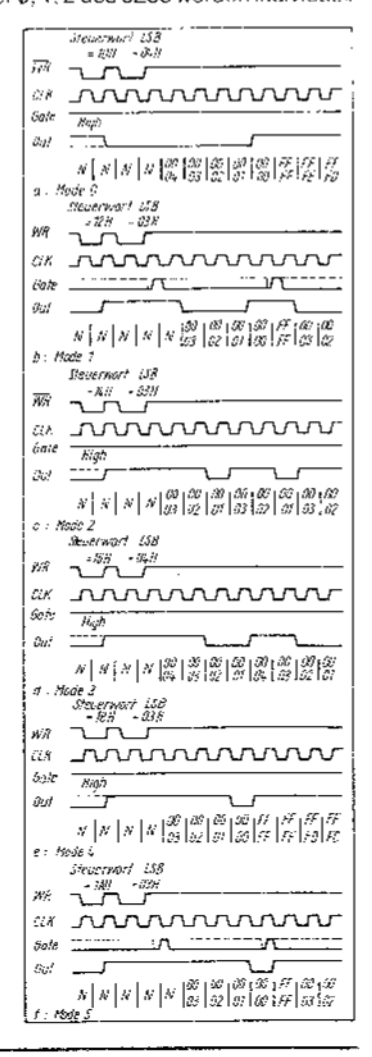
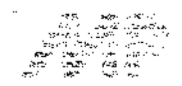


Bild 3.3 Betriebsarten des 8253-PIT
Bild 3.3a Mode 0,
Bild 3.3b Mode 1,
Bild 3.3c Mode 2,
Bild 3.3d Mode 3,
Bild 3.3e Mode 4,
Bild 3.3f Mode 5,



durch Schreiben eines Steuerwortes in das Steuerwortregister initialisiert. Grundsätzlich stehen sechs verschiedene Betriebsarten zur Auswahl:

– ***MODE Ø = Interrupt on Terminal Count***
Nach dem Setzen des Modes geht OUT auf Low (Bild 3.3a). Nachdem die Zählkonstante geladen wurde, erfolgt ein Dekrementieren mit CLK. Beim Erreichen des Zählernullwertes geht OUT auf High und wird erst nach dem Laden einer neuen Zählkonstante oder nach einer neuen Modeeinstellung zurückgesetzt. Ein Low-Signal an GATE sperrt den Zählvorgang.

– ***MODE 1 = Programmable ONE-Shot***
Mit Mode 1 wird eine Monoflopfunktion mit Triggerung am GATE-Eingang realisiert (Bild 3.3b). Eine L/H-Flanke an GATE triggert den Zählvorgang, und OUT geht nach dem nächsten Eingangstaktimpuls auf Low. Beim erreichen des Zählernullwertes wird OUT konstant High. Eine weitere Triggerflanke wiederholt den Vorgang. Eine Triggerflanke bereits vor dem Erreichen des Zählernullwertes lädt den Rückwärtszähler neu und bewirkt wieder eine volle Auszählung ab der Zeit der Triggerung. Wird innerhalb des Dekrementierens ein neuer Zählerwert geladen, so wird zunächst noch der bisherige Zählerwert abgearbeitet, bevor dieser geladene Zählerwert mit erneuter Triggerung zur Wirkung kommt.

– ***MODE 2 = Rate-Generator***
Während des Dekrementierens des Rückwärtszählers verbleibt OUT = High (Bild 3.3c). Bei erreichen des Zählernullwertes wird OUT für eine Periodendauer von CLK gleich Low, und der Vorgang des Rückwärtszählens wiederholt sich in der nächsten Periode. Low-Signal an Gate sperrt den Zählvorgang. Ein Neuladen des Zählregisters während des Dekrementiervorgangs wird erst in der nächsten Periode wirksam. Mode 2 des PIT entspricht etwa der Zeitgeberfunktion des U 857-CTC.

– ***MODE 3 = Square Wave Rate-Generator***
Dieser Mode realisiert die Funktion eines Rechteckwellengenerators und ist im wesentlichen mit dem Mode 2 vergleichbar. Unterschiedlich ist hier das Tastverhältnis der OUT-Signale. Nach dem Laden eines geraden Wertes in das Zählregisters geht OUT für N/2 Taktimpulse zunächst auf High und für die weiteren N/2 Taktimpulse auf Low (Bild 3.3d). Der Rückwärtszähler wird beim Erreichen des Nullwertes automatisch wieder geladen, und der Vorgang wiederholt sich. Ein Low-Signal an GATE sperrt den Zählvorgang. Wenn der Wert im Zählregister ungerade ist, so wird OUT für (N+1)/2 Taktimpulse gleich High und für die folgenden (N−1)/2 Taktimpulse gleich Low. Die Betriebsart Mode 3 wird zur Generierung von Sende- bzw. Empfangstakt für den seriellen Interface-Schaltkreis USART-8251 A verwendet.

– ***MODE 4 = Softwaretriggered Strobe***
Nach dem Setzen des Modes und Laden der Zeitkonstante beginnt der Zählvorgang, und OUT verbleibt auf HIGH (Bild 3.3e). Beim Erreichen des Zählernullwertes wird OUT für eine Taktperiode gleich Low und verbleibt anschließend auf High. Erst nach dem erneuten Laden des Zählregisters wird dieser Vorgang

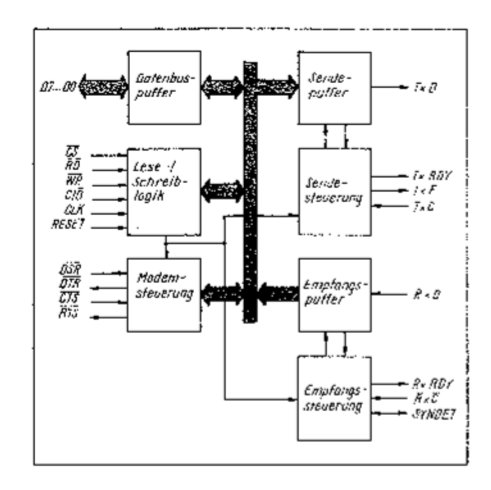
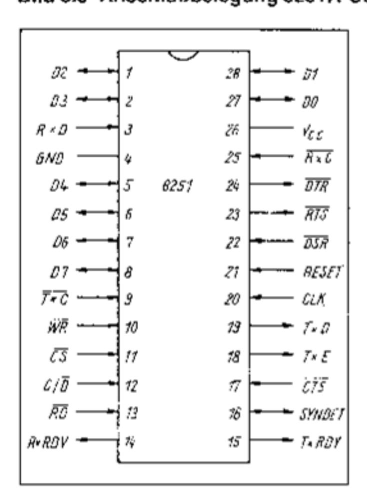


Bild 3.4 Architektur 8251-USART

Bild 3.5 Anschlußbelegung 8251A-USART



Tafel 3.2 8253 Aufbau des Steuerwortes

D7	D6	D5	D4	D3	D2	D1	DØ
SC1	SCØ	RL1	RLØ	M2	M1	MØ	BCD

SC1	SCØ	Zählerauswahl (Select Counter)
Ø	Ø	Zähler Ø
Ø	1	Zähler 1
1	Ø	Zähler 2
1	1	unerlaubt

RL1	RLØ	Lese- oder Ladeoperation (Read/Load)
Ø	Ø	Zähler-Latch-Operation (D3 ... DØ beliebig) Lesen/Laden der Zählerkonstante
Ø	1	Lesen/Laden des niederwertigen Bytes
1	Ø	Lesen/Laden des höherwertigen Bytes
1	1	Lesen/Laden des niederwertigen und danach des höherwertigen Bytes

M2	M1	MØ	Modeauswahl
Ø	Ø	Ø	MODE Ø
Ø	Ø	1	MODE 1
X	1	Ø	MODE 2
X	1	1	MODE 3
1	Ø	Ø	MODE 4
1	Ø	1	MODE 5

BCD	Zählformat
Ø	16-Bit, binär
1	4 Dekaden BCD

wiederholt. Low-Signal an GATE sperrt den Zählvorgang.

– ***MODE 5 = Hardwaretriggered Strobe***
Diese Betriebsart ähnelt Mode 4, nur daß der Zählvorgang durch eine L/H-Flanke an Gate gestartet wird (Bild 3.3f).

3.1.4. Programmierung

In der Grundinitialisierung des Systems wird jeder PIT-Zähler einzeln programmiert.
Im Steuerwort (Tafel 3.2) erfolgt mit D7, D6 die Selektion des gewünschten Zählers. Die Bits D3, D2, D1 legen die Betriebsart fest. Mit den Bits D5, D4 wird entweder eine 1 Byte Zählkonstante oder eine 2 Byte Zählkonstante festgelegt. Das Bit DØ bestimmt das Zählformat. Mit einem OUT-Befehl wird das Steuerwort zur Portadresse Control (A1 = AØ = 1) geschrieben. Danach erfolgt das Laden der Zählkonstante als 1-Byte- oder 2-Byte-Information zur Adresse des im Steuerwort selektierten Zählers mit einer Portadresse nach Tafel 3.1. Somit ergibt sich folgende Sequenz für die Programmierung eines Zählers N:

1. Steuerwort zum Control-Port
2. LSB Zählregister-Byte zum Zähler-N-Port
3. MSB Zählregister-Byte zum Zähler-N-Port

Nach Ausgabe des Steuerwortes können zu beliebigen Zeitpunkten die Zählregister der einzelnen Zähler neu beschrieben werden.
Der aktuelle Zählerstand wird durch IN-Operationen vom Zähler-Port bestimmt.

3.2. Programmierbarer serieller Interfaceschaltkreis 8251 A (USART)

Der USART-Schaltkreis (Universal Synchronous/Asynchronous Receiver/Transmitter) realisiert im Mikrocomputersystem 8086 die serielle Datenübertragung und besitzt folgende wesentliche Leistungsmerkmale:

- synchrone und asynchrone Übertragung
- doppelt gepufferter Sender und Empfänger
- Übertragung von Zeichen im Format von 5 bis 8 Bits
- Übertragungsgeschwindigkeit bis zu 64 kBaud
- Synchron-Mode:
  – automatische Sync-Einfügung
  – interne oder externe Zeichensynchronisation
- Asynchron-Mode:
  – Clockrate: × 1, × 16, × 64
- automatische Breakerkennung
- automatische Fehlererkennung

3.2.1. Architektur

Die USART-Funktionseinheiten (Bild 3.4) haben folgende Aufgaben:
Über den Datenbuspuffer, als Schnittstelle zum Systemdatenbus, werden Steuerworte/Daten transferiert:

– Schreiben des Modesteuerwortes
– Schreiben des Befehlssteuerwortes
– Schreiben der zu sendenden Daten
– Lesen der zu empfangenden Daten
– Lesen der Statusinformation

Die Lese/Schreiblogik leitet aus den Signalen Chip-Select $\overline{CS}$, Clock CLK, Read $\overline{RD}$, Write $\overline{WR}$, Control/Data C/$\overline{D}$ und RESET Steuersignale für die Schaltkreisfunktionen ab.

Der Sendepuffer erhält die Daten im parallelen Format vom Datenbuspuffer und konvertiert diese in einen seriellen Datenstrom, der zur Ausgabe geführt wird. Entsprechend dem Übertragungsmodus werden zusätzliche Zeichen oder Bits für Kontrollfunktionen eingefügt.
Der Empfangspuffer empfängt einen seriellen Datenstrom und wandelt diesen in ein paralleles Format um. Dabei werden die eingefügten Kontroll-Zeichen/Bits ausgewertet.

3.2.2. Pinbelegung

Der USART 8251A im 28-Pin-DIL-Gehäuse (Bild 3.5) hat die Anschlußbelegung mit folgenden ***Funktionen:***

D7–D0 8-Bit-Datenbus bidirektional, 3-state

C/$\overline{\text{D}}$ Control/Data, Eingang
Steuerleitung zur Selektion von Steuer- oder Datenworten
C/$\overline{\text{D}}$ = High Control/Status
C/$\overline{\text{D}}$ = Low Data

$\overline{\text{RD}}$ Read

$\overline{\text{WR}}$ Write

$\overline{\text{CS}}$ Chip-Select

Diese Funktionen sind mit denen des 8253-PIT identisch.

CLK Systemtakt, Eingang
Der USART-Systemtakt (max. 3 MHz) ist mit dem Peripherietakt PCLK des Clockgenerators 8284A zu verbinden.

RESET Rücksetzen, Eingang high-aktiv

$\overline{\text{TxC}}$ Sendertakt (Transmitter Clock), Eingang
max. Frequenz:
1 × Bitrate 64 kHz
16 × Bitrate 310 kHz
64 × Bitrate 615 kHz

TxD serielle Sendedaten (Transmitter Data), Ausgang

$\overline{\text{RxC}}$ Empfängertakt (Receiver Clock), Eingang
max. Frequenz: vgl. **$\overline{\text{TxC}}$**

RxD serielle Empfangsdaten (Receiver Data), Eingang

TxRDY Sender bereit (Transmitter Ready), Ausgang
Dieser Ausgang liefert der CPU die Information, daß der USART zur Datenübernahme in Form eines OUT-Befehls bereit ist.
Beim Schreiben eines Zeichens in den USART erfolgt ein automatisches Rücksetzen von TxRDY.
TxRDY kann zur Interrupt-Auslösung am Interrupt-Controller 8259A verwendet werden, um in der Interrupt-Service-Routine eine Daten-Schreiboperation vorzunehmen.

Tafel 3.3 8251A Steuersignale (mit $\overline{\text{CS}}$ = 0)

$\overline{\text{RD}}$	$\overline{\text{WR}}$	C/$\overline{\text{D}}$	Funktion
0	1	0	Daten vom USART zum Systembus (Portlesen)
0	1	1	Statuslesen vom USART
1	0	0	Daten vom Systembus zum USART (Portschreiben)
1	0	1	USART-Steuerwort schreiben

RxRDY Empfänger bereit (Receiver Ready), Ausgang
Dieser Ausgang liefert der CPU die Information, daß der USART-Empfangspuffer ein Zeichen enthält und damit die Bereitschaft zur Übertragung in die CPU in Form eines IN-Befehls vorliegt.
RxRDY wird mit $\overline{\text{RD}}$ zurückgesetzt. RxRDY kann über den Interrupt-Controller eine Interrupt-Service-Routine zum USART-Lesen einleiten.

$\overline{\text{DSR}}$ Data Set Ready, Eingang, low-aktiv
Steuerleitung für den Modembetrieb
$\overline{\text{DSR}}$ kann im Bit 7 des Statusregisters gelesen werden.
$\overline{\text{DSR}}$ hat keinen Einfluß auf das Senden/Empfangen von Daten.

$\overline{\text{DTR}}$ Data Terminal Ready, Ausgang low-aktiv
$\overline{\text{DTR}}$ wird durch Bit 1 im Befehlssteuerwort gesetzt/rückgesetzt.

$\overline{\text{RTS}}$ Request To Send, Ausgang low-aktiv
Steuerleitung für den Modembetrieb
$\overline{\text{RTS}}$ wird durch Bit 5 im Befehlssteuerwort gesetzt/rückgesetzt.

$\overline{\text{CTS}}$ Clear To Send, Eingang, low-aktiv
$\overline{\text{CTS}}$ = Low startet das Senden der seriellen Daten an TxD, wenn das Bit 0 im Befehlssteuerwort (Sendefreigabe) gleich „1" gesetzt wurde. Ein Software-Start des Sendens ohne $\overline{\text{CTS}}$ = Low ist beim USART nicht möglich.

TxE Sendepuffer leer (Transmitter Empty), Ausgang, high-aktiv
TxE liefert High-Signal, wenn der Parallel-Serien-Converter im Transmitter leer ist. TxE wird automatisch gleich Low, wenn ein Byte in den Sendepuffer geschrieben wurde.

SYNDET Ein-/Ausgang
Synchronisations- oder Breakerkennung
Synchron-Mode (SYNC-Detect)
In dem internen Synchronisationsmode notiert der Ausgang SYNDET gleich High ein gefundenes SYNC-Zeichen. Im Bisync-Mode geht der Ausgang SYNDET bei der Erkennung des letzten Bits des zweiten SYNC-Zeichens auf High. Bei einer Status-Leseoperation wird SYNDET automatisch zurückgesetzt. Wenn externer SYNC-Mode programmiert ist, wirkt SYNDET als Eingang, und ein positiver Pegelübergang an SYNDET startet die Zeichenerfassung am seriellen Dateneingang RxD.
Asynchron-Mode (Break Detect)
Ausgang
High-Signal wird ausgegeben, wenn alle Zeichen des seriellen Datenprotokolls (Start-Bit, Daten-Bit, Paritätsbit, 1 Stop-Bit) als 0 erkannt werden.

Vcc Betriebsspannung + 5 V

GND Masseanschluß

Tafel 3.4 8251A Aufbau des Modesteuerwortes

Asynchron-Mode

D7	D6	D5	D4	D3	D2	D1	D0
S2	S1	EP	PEN	L2	L1	B2	B1

S2	S1	Anzahl der Stopbits
0	0	nicht erlaubt
0	1	1 Bit
1	0	1 1/2 Bits
1	1	2 Bits

EP	Paritätsgenerierung/-test (Even Parity Generation/Check)
0	ungerade
1	gerade

PEN	Paritätsfreigabe (Parity enable)
0	gesperrt
1	freigegeben

L2	L1	Zeichenlänge
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

B2	B1	Bitratefaktor
0	0	nicht erlaubt (Synchron-Mode)
0	1	×1
1	0	×16
1	1	×64

Synchron-Mode

D7	D6	D5	D4	D3	D2	D1	D0
SCS	ESD	EP	PEN	L2	L1	0	0

SCS	Anzahl der SYNC-Zeichen Single-Character Sync
0	zwei Sync-Zeichen
1	ein Sync-Zeichen

ESD	externe Synchronisationserkennung (External Sync Detect)
0	Intern-Sync
1	Extern-Sync

EP	Paritätsgenerierung/-test (Even Parity Generation/Check)
0	ungerade
1	gerade

PEN	Paritätsfreigabe (Parity enable)
0	gesperrt
1	freigegeben

L2	L1	Zeichenlänge
0	0	[illegible]
0	1	[illegible]
1	0	7 Bits
1	1	[illegible] Bits

3.2.3. Programmierung

Die Funktion des USART wird durch Programmierung mit zwei Steuerwörtern, dem Modesteuerwort und dem Befehlssteuerwort, festgelegt. Im Modesteuerwort (Tafel 3.4) werden folgende Vereinbarungen getroffen:
- Synchron- oder Asynchron-Mode
- Bitratenfaktor im Asynchron-Mode (x1, x16, x64)
- Zeichenlänge (5 ... 8 Bit)
- Paritätskontrolle
- Anzahl der Stopbits (1, 1½, 2)
- Synchronisations-Steuerung (intern, extern, Double SYNC-Character, Single SYNC-Character)

Das Befehlssteuerwort (Tafel 3.5) legt folgende Parameter fest:
- Sender-/Empfängerfreigabe
- Setzen der Modemsteuersignale DTR, RTS
- Reset-Funktionen

Die Programmierung von Modesteuerwort und Befehlssteuerwort muß in einer definierten Reihenfolge vorgenommen werden. Nach dem internen oder externen Rücksetzen des 8251A wird das erste Steuerwort als Modesteuerwort interpretiert, alle folgenden Steuerwörter werden als Befehlssteuerwörter erkannt. Das interne Rücksetzen kann im Befehlssteuerwort (Bit 6 = 1) festgelegt werden. Die Betriebsarten des USART entsprechen im wesentlichen den bekannten Prinzipien für Asynchron-/Synchron-Mode. Im SYNC-Mode besitzt der USART eine geringere Leistungfähigkeit als der Schaltkreis U 856-SIO.

Tafel 3.5 8251A Aufbau des Befehlssteuerwortes

D7	D6	D5	D4	D3	D2	D1	D0
EH	IR	RTS	ER	SBRK	RxEN	DTR	TxEN

EH	Suchen von Sync-Zeichen (Error Hunt) (nur im Synchron-Mode)
0	Suchbetrieb stop
1	Suchbetrieb freigegeben
IR	internes RESET
0	Verbleiben im Befehlssteuerwort
1	Rückkehr zum Modesteuerwort
RTS	Sendeanforderung (Request To Send)
0	Pin $\overline{RTS}$ – High (inaktiv)
1	Pin $\overline{RTS}$ – Low (aktiv)
ER	Fehler-Reset (Error Reset)
0	keine Beeinflussung der Fehler-Flags
1	Rücksetzen der Fehlerflags PE, OE und FE
SBRK	Sendeunterbrechung (Send Break Character)
0	normale Operation
1	PIN TxD – Low
RxEN	Empfängerfreigabe (Receive Enable)
0	Empfang gesperrt
1	Empfänger freigegeben
DTR	Datenendstelle bereit (Data Terminal Ready)
0	Pin $\overline{DTR}$ = High (inaktiv)
1	Pin $\overline{DTR}$ – Low (aktiv)
TxEN	Senderfreigabe (Transmit Enable)
0	Senden gesperrt
1	Sender freigegeben

● **Asynchron-Mode**

Das Datenformat ist im Bild 3.6 dargestellt.

Senden

Im passiven Zustand liegt das High-Signal am Ausgang TxD. Das Senden beginnt bei CTS = Low mit einem Startbit, dann folgen die Datenbits ab D0 bis zur programmierten Zeichenlänge. Den Daten folgt ein Paritätsbit, falls dieses im Modesteuerwort freigegeben wurde. Den Abschluß bildet die programmierte Anzahl der Stopbits. Wenn der Zeichenpuffer leer ist und keine Breakausgabe im Befehlssteuerwort programmiert wurde, geht TxD auf High. Das Ende des Sendens wird mit einem High-Pegel am Pin TxRDY bzw. im Statusbit notiert. Die seriellen Daten werden mit der fallenden Flanke von $\overline{TxC}$, geteilt durch den programmierten Bitratefaktor, gesendet.

Empfangen

Ein konstanter High-Pegel an RxD wird als Ruhezustand interpretiert. Eine fallende Flanke an RxD kennzeichnet den Beginn des Startbits. Der Pegel an RxD wird mit der steigenden Flanke von $\overline{RxC}$, geteilt durch den Bitratefaktor, abgefragt und Datenbits, Paritätsbit (falls programmiert) und Stopbits entsprechend Bild 3.4 in einen Serien-Parallel-Wandler übernommen. Beträgt die Zeichenlänge weniger als 8 Bits, so werden für die nicht vorhandenen Bits Nullen eingefügt. Bei einem Paritätsfehler wird das Parity-Error-Flag gesetzt. Falls nach dem Paritätsbit ein Low-Pegel als Stopbit festgestellt wird, so erfolgt ein Setzen des Framing-Error-Flags. Unabhängig von der Anzahl der programmierten Stopbits fordert der Empfänger nur ein Stopbit. Ist der Datenbuspuffer geladen, so wird das Pin bzw. Statusbit RxRDY gesetzt. Damit wird der CPU mitgeteilt, daß ein Byte zur Abholung in Form eines IN-Befehls bereitsteht. Wenn dieses von der CPU nicht eingelesen wurde, so erfolgt ein Überschreiben mit dem nächsten empfangenen Zeichen, dabei wird das Overrun-Error-Flag gesetzt.

● **Synchron-Mode**

Das Datenformat ist im Bild 3.7 dargestellt.

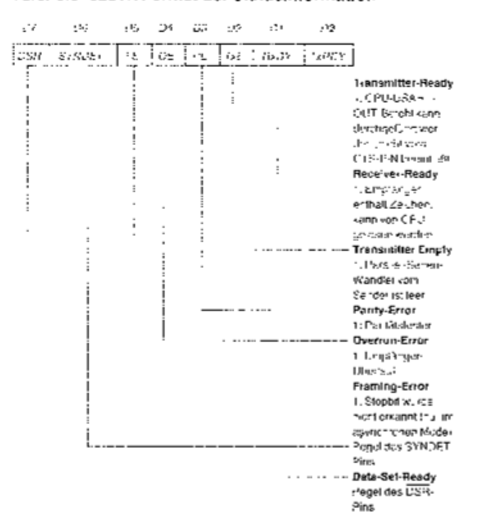
Senden

Im Synchron-Mode sind die Daten von SYNC-Zeichen eingeschlossen. Mit $\overline{CTS}$ = Low werden die Daten mit der fallenden Flanke von $\overline{TxC}$ an TxD hinausgeschoben. Nachdem der Sendepuffer leer ist, werden automatisch Sync-Zeichen in den Datenstrom eingefügt, bis ein neues Byte in den 8251A geschrieben wurde. Der Ausgang Transmitter Empty TxE = High teilt der CPU mit, daß der Sendepuffer leer ist, und TxE wird erst mit dem nächsten OUT-Befehl auf Low zurückgesetzt.

Empfangen

Die Zeichensynchronisation kann intern oder extern ausgeführt werden. Im Befehlssteuerwort sollte der Suchbetrieb mit D7 = 1 freigegeben werden. Mit der steigenden Flanke von $\overline{RxC}$ werden die an RxD liegenden Pegel übernommen. Wenn die Sync-Zeichen erkannt wurden, beendet der USART den Suchbetrieb und ist synchronisiert. Der Anschluß SYNDET wird anschließend auf High gesetzt und beim Statuslesen automatisch zurückgesetzt. Im externen Synchronisationsmode wird die Synchronisation durch den High-Übergang am Eingang SYNDET bis zum nächsten $\overline{RxC}$-Zyklus gestartet. Paritäts- und Überlauffehler werden wie beim asynchronen Empfang überprüft.

Tafel 3.6 8251A Format der Statusinformation



Tafel 3.7 Pollinggesteuerte Ein-/Ausgabe für USART

```
INPUT:  IN    AL,USART_CONTROL  ;Lesen Statusregister
        TEST  AL,2              ;Prüfen RxDY = 1
        JZ    INPUT
        IN    AL,USART_DATA     ;gültiges Zeichen
                                ;vorhanden, Daten lesen
        PUSH  AX                ;in AL auszugebendes
                                Byte
OUTPUT: IN    AL,USART_CONTROL  ;Lesen Statusregister
        TEST  AL,1              ;prüfen TxRDY = 1
        JZ    OUTPUT
        POP   AX                ;Sendepuffer leer
        OUT   USART_DATA,AL     ;Daten schreiben
```

3.2.4. Statusregister

Das Statusregister enthält Informationen über die Sende-/Empfangs- und Fehlerbedingungen und den logischen Pegel einiger Kommunikationssignale (Tafel 3.6). Das Statusregister kann durch IN-Befehl von der USART-Control-Adresse (C/$\overline{D}$ = 1) gelesen werden.

Die Statusbits $\overline{DSR}$, SYNDET, TxE und RxRDY geben den aktuellen Logikpegel der entsprechenden Anschlußpins wieder. TxRDY informiert, daß ein neues Zeichen von der CPU in den Datenbuspuffer geschrieben werden kann und ist unabhängig von einer programmierten Sendefreigabe (TxEN-Bit im Befehlssteuerwort) und vom $\overline{CTS}$-Pin. Die Statusbits

FE Framing Error: Stopbit-Fehler,

OE Overrun Error: Zeichen wurde von der CPU nicht abgeholt,

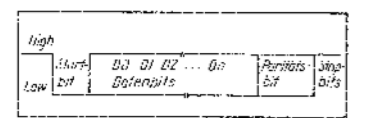


Bild 3.6 Datenformat im Asynchron-Mode

Bild 3.7 Datenformat im Synchronmode

Sync-Zeichen 1 | Sync-Zeichen 2 | Daten (5-8 Bits/Zeichen) | Daten

PE Paritätsfehler
dienen als Fehlerflags.
Für die Polling-gesteuerte Eingabe/Ausgabe werden die Statusbits RxRDY und TxRDY verwendet (Tafel 3.7).

3.3. Programmierbarer paralleler Interfaceschaltkreis 8255A (PPI)

Der PPI-Schaltkreis 8255A (Programmable Peripheral Interface) realisiert im Mikroprozessorsystem 8086 die parallele Ein-/Ausgabe und ist durch folgende Leistungsmerkmale gekennzeichnet:
- 3 programmierbare Ein-/Ausgabeports; Port A, B, C
- 3 Betriebsarten
Mode 0: Basic Input/Output
Mode 1: Strobed Input/Output
Mode 2: Strobed Bidirectional Bus
- Einzelbit-Set/Reset-Operation an Port C
- Interruptauslösung in Mode 1 und 2 in Verbindung mit Interrupt-Controler 8259A
- kein Systemtakt erforderlich

3.3.1. Architektur

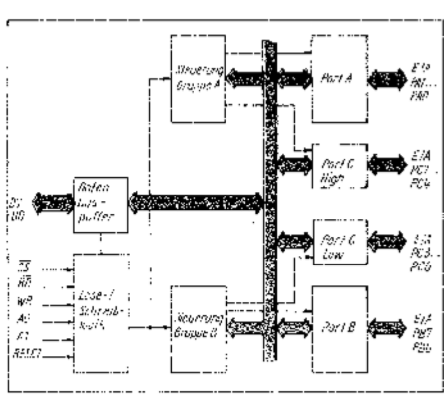
Die Architektur des PPI zeigt Bild 3.8. Die Funktionen des Datenbuspuffers und der Lese-/Schreiblogik entsprechen denen des 8253 PIT. Die Adressen für Port A, B, C und für das Steuerwort sind in Tafel 3.8 dargestellt. Auf der Peripherieseite werden die Ports in zwei Gruppen eingeteilt:
Gruppe A: 8-Bit-Port A PA7 – PA0
4-Bit-Port C PC7 – PC4
Gruppe B: 8-Bit-Port B PB7 – PB0
4-Bit-Port C PC3 – PC0
Die Programmierung mit Modeeinstellung erfolgt für beide Gruppen zusammen in einem Steuerwort.
Die Ports haben folgende Eigenschaften:

Port A
Das Port A wird vorzugsweise als 8-Bit-Ein-/Ausgabe-Latch/Puffer verwendet. Der bidirektionale Betrieb ist nur mit Port A möglich.

Port B
Port B wird vorzugsweise als 8-Bit-Ausgangs-Latch/Puffer oder als 8-Bit-Eingangslatch verwendet.

Port C
Port C wird in zwei 4-Bit-Ports aufgeteilt:
PC7 ... PC4
PC3 ... PC0
Die Teilports können einzeln als Ausgangs-Latch/Puffer oder Eingangspuffer in Mode 0 programmiert werden. In den Handshaking-Betriebsarten werden einige Port-C-Leitungen als Steuersignale den Ports A und B zugeordnet.

Bild 3.8 Architektur 8255A-PPI



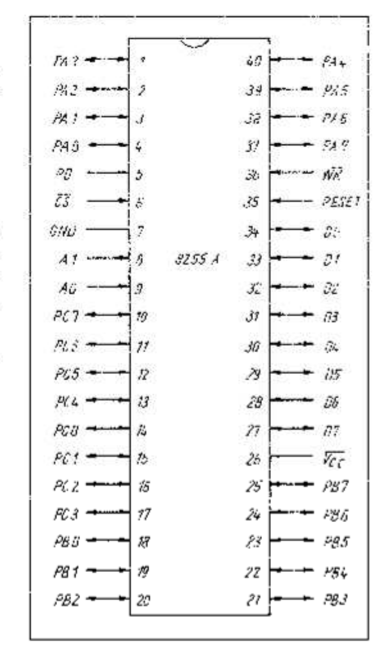


Bild 3.9 Anschlußbelegung 8255A-PPI

Tafel 3.8 8255A Portadressen

A1	A0	Port
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Steuerwort (nur Schreiben)

Tafel 3.9 8255A Aufbau des Steuerwortes

	D7	D6	D5	D4	D3	D2	D1	D0
		Gruppe A				Gruppe B		
Modesteuerung →	S=1	M2	M1	A	CH	M0	B	CL
Einzel-Bit-Ausgabe Port C →	S=0	X	X	X	C2	C1	C0	S/R

Modesteuerung (S = 1)

M2	M1	Modeauswahl
0	0	Mode 0
0	1	Mode 1
1	X	Mode 2

A Auswahl Port A, PA7 ... PA0 ⟍ als Ausgang: 0
CH Auswahl Port-C-High PC7 ... PC4 ⟋ oder Eingang: 1

M0	Modeauswahl
0	Mode 0
1	Mode 1

B Auswahl Port B, PB7 ... PB0 ⟍ als Ausgang: 0
CL Auswahl Port-C-Low PC3 ... PC0 ⟋ oder Eingang: 1

Einzel-Bit-Ausgabe für Port C (S = 0)

C2	C1	C0	Bitauswahl für Port C
0	0	0	Portleitung PC0
0	0	1	Portleitung PC1
0	1	0	Portleitung PC2
0	1	1	Portleitung PC3
1	0	0	Portleitung PC4
1	0	1	Portleitung PC5
1	1	0	Portleitung PC6
1	1	1	Portleitung PC7

S/R	Setz-/Rücksetzfunktion für PC0 ... PC7
0	rücksetzen
1	setzen

3.3.2. Pinbelegung

Der PPI 8255A wird in einem 40-poligen DIL-Gehäuse gefertigt. Die Anschlüsse (Bild 3.9) haben folgende Funktionen:

D7–D0 8-Bit-Datenbus bidirektional, 3-state
$\overline{RD}$ $\overline{WR}$ $\overline{CS}$ entsprechen den Funktionen des 8253-PIT.
RESET Rücksetzen, Eingang high-aktiv Löscht das Steuerwortregister, und alle Portanschlüsse werden als Eingang geschaltet.
A0 A1 Adreßeingänge zur Portauswahl Diese Signalleitungen adressieren die E/A-Ports und das Steuerwortregister nach Tafel 3.8. A0 und A1 werden zum 8086-System-Adreßbus AB1 und AB2 geführt.
PA0–PA7 Port A
$U_{OL} \leqq 0{,}45$ V, $I_{OL} \leqq 1{,}7$ mA
$U_{OH} \geqq 2{,}40$ V, $I_{OH} \geqq -200\,\mu$A
PB0–PB7 Port B
PC0–PC7 Port C
Port B,C Darlington-Strom
$R_{ext} = 750$ Ohm
$U_{ext} = 1{,}5$ V
$I = -1{,}0 \ldots 4{,}0$ mA
Vcc Betriebsspannung –5 V
GND Masseanschluß

3.3.3. Programmierung

Die Programmierung des 8255A erfolgt mit einem Steuerwort (Tafel 3.9) in zwei Varianten.
Modesteuerwort (D7 = 1):
Programmierung der Ports A, B, C in den Betriebsarten 0, 1, 2
Einzel-Bit-Ausgabe (D7 = 0):
Set-/Reset-Ausgaben für jedes Bit von Port C einzeln.

Modesteuerwort
Die Modeeinstellung erfolgt getrennt für die Gruppen A und B, dabei ergeben sich für die Ports A, B, C folgende Varianten:
Port A: Mode 0 oder 1 oder 2
Port B: Mode 0 oder 1
Port C: Mode 0

– MODE 0 = Basic Input/Output
In Mode 0 werden Ein-/Ausgabe-Operationen ohne Quittung vorgenommen. Wenn alle Ports des 8255A in Mode 0 programmiert werden (M2 = M1 = M0 = 0), so stehen folgende Ein-/Ausgaben zur Verfügung:
- zwei 8-Bit-Ports für Byte Ein-/Ausgabe (Port A, B)
- zwei 4-Bit-Ports für Halbbyte-Ein-/Ausgabe (Port C)
Die Portausgänge sind gelatcht, die Porteingänge sind nicht gelatcht.

– MODE 1 = Strobed Input/Output
Die Ports A und B arbeiten als Ein- oder Ausgabeport im Quittungsbetrieb in Verbindung mit dem Port C, welches die Handshakingsignale verwaltet. In den quittungsgesteuerten Ein-/Ausgaben kann vom 8255A eine Interrupt-Anforderung an den Programmierbaren-Interrupt-Controller 8259A angemeldet werden. Die Interruptfreigabe des 8255A wird

durch ein internes INTE-Flip-Flop organisiert, welches durch Einzelbit-Set/Reset-Operationen von Port C beeinflußt wird.
Bit-Set – Interrupt-Freigabe
Bit-Reset = Interrupt-Sperre
Die Steuersignale haben im Mode 1 folgende ***Funktion:***

Steuersignale für Port-Input-Operationen

$\overline{\text{STB}}$ Strobe, Eingang, low-aktiv
$\overline{\text{STB}}$ – Low lädt die Daten in das Port-Eingangslatch.

IBF Input Buffer Full, Ausgang low-aktiv
High notiert, daß Daten in das Eingangslatch geladen worden sind und stellt somit ein Bestätigungssignal dar. IBF wird mit $\overline{\text{STB}}$ = Low gesetzt und mit der Rückflanke von $\overline{\text{RD}}$ zurückgesetzt.

INTR Interrupt Request, Ausgang, high-aktiv
INTR wird gesetzt, wenn nach dem Latchen der Port-Eingabedaten $\overline{\text{STB}}$ und IBF gleich High sind. In der daraufhin eingeleiteten Interrupt-Service-Routine mit Lesen der Port-Eingabe-Daten wird INTR mit der Vorderflanke von $\overline{\text{RD}}$ zurückgesetzt. Die INTE-Flip-Flops von Port A und B werden kontrolliert durch:
INTE Port A: Bit Set/Reset PC4
INTE Port B: Bit Set/Reset PC2

Steuersignale für Port-Output-Operationen

$\overline{\text{OBF}}$ Output Buffer Full, Ausgang low-aktiv
$\overline{\text{OBF}}$ aktiv = Low notiert, daß die CPU Daten in das Port geschrieben hat, die an den Portausgängen gültig bereitstehen. Diese Aktivierung von $\overline{\text{OBF}}$ erfolgt nach der Rückflanke von $\overline{\text{WR}}$.

$\overline{\text{ACK}}$ Acknowledge Input, Eingang, low-aktiv
$\overline{\text{ACK}}$ gleich Low notiert, daß die Peripherie vom 8255A die gültigen Daten übernommen hat. Mit $\overline{\text{ACK}}$ = Low wird $\overline{\text{OBF}}$ wieder inaktiv = High.

INTR Interrupt Request, Ausgang, high-aktiv
Mit $\overline{\text{OBF}}$ = High und $\overline{\text{ACK}}$ = High löst ein aktives INTR = High einen Interrupt aus, der in der Interrupt-Service-Routine zum Schreiben neuer Port-Ausgabe-Daten führt.
Mit der Vorderflanke von $\overline{\text{WR}}$ wird INTR inaktiv gleich Low, und am Ende des Bestätigungs-Zyklus mit der Rückflanke von $\overline{\text{ACK}}$ wird INTR wieder aktiv gleich High und damit ein neuer Interrupt ausgelöst.
Die INTE-Flip-Flops von Port A und Port B werden kontrolliert durch:
INTE Port A: Bit Set/Reset PC6
INTE Port B: Bit Set/Reset PC2

Bild 3.10 Varianten von Mode 1; Strobed Input/Output
Bild 3.10a Mode 1, Port A – Input
Bild 3.10b Mode 1, Port A – Output
Bild 3.10c Mode 1, Port B – Input
Bild 3.10d Mode 1, Port B – Output

Tafel 3.10 8255A Format der Statusinformation bei MODE 1 (IN-PORT C)

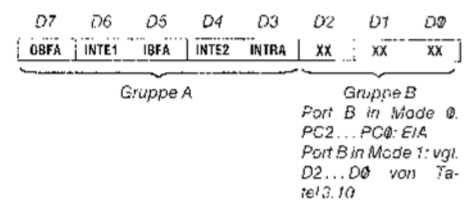
Für Port-Eingabe:

D7	D6	D5	D4	D3	D2	D1	D0
E/A	E/A	IBFA	INTEA	INTRA	INTEB	IBFB	INTRB
Gruppe A					Gruppe B		

Für Port-Ausgabe:

D7	D6	D5	D4	D3	D2	D1	D0
OBFA	INTEA	E/A	E/A	INTRA	INTEB	OBFB	INTRB
Gruppe A					Gruppe B		

Tafel 3.11 8255A Format der Statusinformation bei MODE 2 (IN-Port C)

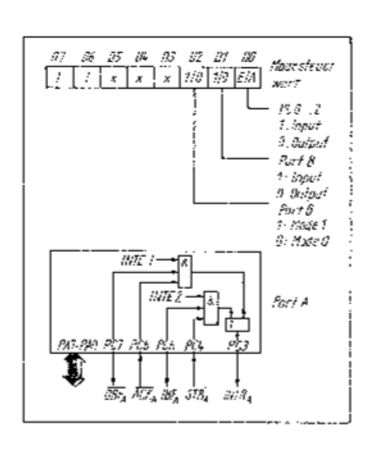


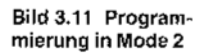
D7	D6	D5	D4	D3	D2	D1	D0
OBFA	INTE1	IBFA	INTE2	INTRA	XX	XX	XX
Gruppe A					Gruppe B		

Port B in Mode 0: PC2 ... PC0: E/A
Port B in Mode 1: vgl. D2 ... D0 von Tafel 3.10

Die Varianten der Programmierung in Mode 1, getrennt nach Port A und B, jeweils für Eingang/Ausgang zeigen die Bilder 3.10a ... d. Informationen über den Zustand der Bestätigungssignale IBF, $\overline{\text{OBF}}$, des Interrupt-Freigabe-Flip-Flops INTE und der Interrupt-Anforderung INTR erhält man durch Lesen eines Status-Wortes von Port C (Vergl. Tafel 3.10).

– ***MODE 2 = Strobed Bidirectional Bus I/O***
Die bidirektionale Port-Ein-/Ausgabe im Quittungsbetrieb wird nur über Port A realisiert. Am Port C befinden sich die Steuersignale für




Bild 3.11 Programmierung in Mode 2

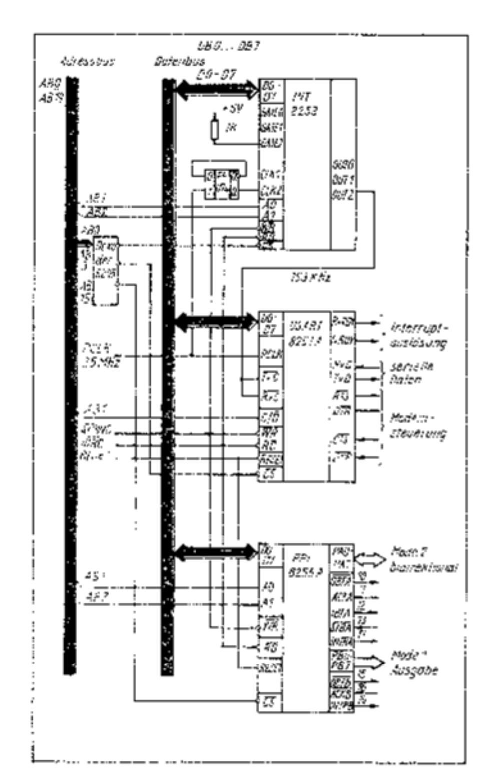


Bild 3.12 Interface-Schaltkreise im System 8086

die Ausgabe $\overline{\text{OBFA}}$ und $\overline{\text{ACKA}}$ und für die Eingabe $\overline{\text{STBA}}$ und IBFA und das Interrupt-Anforderungssignal INTRA. Die Interrupt-Freigabe-Flip-Flops werden wiederum mit Bit-Setz-/Rücksetzfunktionen von Port C beeinflußt.
Ausgabe: Bit Set/Reset PC6 (INTE1)
Eingabe: Bit Set/Reset PC4 (INTE2)
Bild 3.11 zeigt den Aufbau des Modesteuerwortes, die zugehörige Pinbelegung von Port A und die Steuerleitungen an Port C. Wenn Mode 2 programmiert wurde, kann für Port B noch gewählt werden:
Mode 1: Strobed Input/Output mit den Handshaking-Signalen an PC0, PC1, PC2 nach Bild 3.10c und 3.10d
Mode 0: Basic Input/Output für PB0 ... PB7 mit frei wählbaren E/A-Leitungen an PC0 ... PC2.
Das Format der Statusinformation von Mode 2 ist in Tafel 3.11 dargestellt.

Tafel 3.12 Initialisierung der Interfaceschaltkreise

```
;USART – INITIALISIERUNG
      MOV   DX,ØDAH   ;Portadresse USART, Control
      MOV   AL,ØØ     ;Vor-Reset
      OUT   DX,AL
      MUL   AL        ;Delay
      MUL   AL        ;
      OUT   DX,AL
      MUL   AL
      MUL   AL
      OUT   DX,AL
      MUL   AL
      MUL   AL
      MOV   AL,4ØH    ;Reset
      OUT   DX,AL
      MUL   AL
      MUL   AL
      MOV   AL,5AH    ;MODE-Steuerwort
                      ;Bitratefaktor 16, asynchron,
                      ;7 Bit/Charakter,
                      ;ungerade Parität frei-
                      ;gegeben, 1 Stopbit
      OUT   DX,AL
      MUL   AL
      MUL   AL
      MOV   AL,17H    ;Befehlssteuerwort
                      ;kein internes Reset,
                      ;RTS inaktiv,
                      ;Rücksetzen der Fehlerflags,
                      ;Empfängerfreigabe,
                      ;DTR aktiv, Senderfreigabe
      OUT   DX,AL
      MUL   AL
      MUL   AL
;PIT – INITIALISIERUNG
      MOV   DX,ØD6H   ;Portadresse PIT, Control
      MOV   AL,ØB6H   ;Zähler 2, 2 Byte-Zähler-
                      ;konstante,
                      ;MODE 3, 16 Bit-binaer
      OUT   DX,AL
      MOV   DX,ØD4H   ;Portadresse Zähler 2
      MOV   AL,08     ;Zählkonstante LSB
      OUT   DX,AL
      MOV   AL,ØØ     ;Zählkonstante MSB
      OUT   DX,AL     ;CLK: 1,23 MHz
                      ;OUT: 153 kHz
;PPI – INITIALISIERUNG
      MOV   DX,ØCEH   ;Portadresse PPI, Control
      MOV   AL,ØC4H   ;MODE-Steuerung
                      ;MODE 2 für Port A
                      ;MODE 1 für Port B, Portaus-
                      ;gabe:
                      ;(Bit 7, Bit 6, Bit 2 – 1;
                      ;Bit 1 = 0;
                      ;Bit 5, Bit 4, Bit 3, Bit 0:
                      ;beliebig = 0)
      OUT   DX,AL
      MOV   AL,Ø5     ;Interruptfreigabe für Port-
                      ;eingabe B
                      ;INTEB = PC2: set
      OUT   DX,AL
      MOV   AL,Ø9     ;Interruptfreigabe für Port-
                      ;ausgabe A
                      ;INTE2 = PC4: set
      MOV   DX,ØD8H   ;Portadresse USART, Data
                      ;Dateneingabe
      IN    AL,DX     ;≙ RxRDY Statusbit löschen
```

Einzel-Bit-Ausgabe

Jedes der 8 Bits von Port C kann einzeln gesetzt oder rückgesetzt werden durch Einzel-Bit-Ausgabe-Operationen von Port C nach Tafel 3.9 mit D7 – Ø. Wenn Port C in den Handshaking-Betriebsarten Mode 1 oder 2 als Status/Control-Wort für Port A oder B verwendet wird, kann ein gezieltes Freigeben/Sperren der Interrupt-Logik mit INTE-A, INTE-B (Tafel 3.10) und INTE-1, INTE-2 (Tafel 3.11) erfolgen. Die anderen Statusbits sind nicht durch Einzel-Bit-Ausgaben beeinflußbar.

3.4. Anwendungsbeispiel

In Bild 3.12 ist eine applikative Lösung für die Einbindung der Interfaceschaltkreise 8253, 8251A und 8255A in das Mikrorechnersystem 8086 dargestellt. Ein entsprechendes Programmierbeispiel für deren Initialisierung zeigt Tafel 3.12. Der Kanal 2 des PIT wird zur Erzeugung der Taktsignale für eine asynchrone Datenübertragung mit 9,6 kBaud und 7 Bit-ASCII-Zeichen genutzt. Bei der Taktversorgung des PIT 8253 ist darauf zu achten, daß die maximale Eingangsfrequenz von 2,0 MHz nicht überschritten wird und deshalb bei der Verwendung des Signals PCLK eine 2:1-Frequenzteilung vorgenommen werden muß. Der PPI ist für den Handshaking-Betrieb der Ports A und B initialisiert, wobei mit Port A ein bidirektionaler Betrieb und mit Port B eine Dateiausgabe in Mode 1 vorgesehen ist. Um eine sichere Initialisierung der Interface-Schaltkreise im System 8086 zu gewährleisten, müssen jeweils zwischen die Steueranweisungen zwei MUL AL-Befehle zur Verzögerung programmiert werden (beim USART zwingend erforderlich). Bei der softwaremäßigen RESET-Erzeugung für den USART wird durch die Ausgabe von Vor-Resets ein sicheres Rücksetzen gewährleistet.

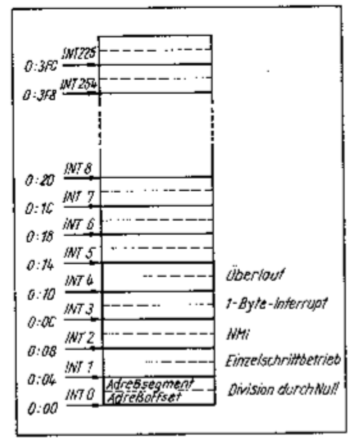


Bild 4.1 Interrupttabelle

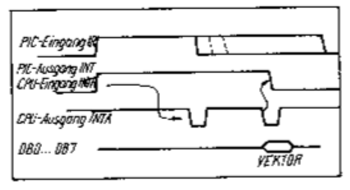


Bild 4.2 Signalverlauf zwischen PIC und CPU

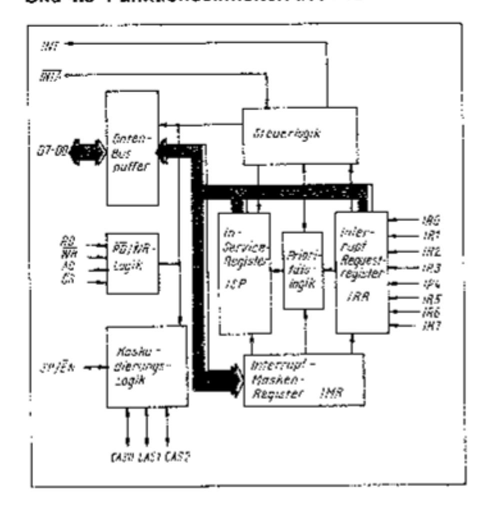


Bild 4.3 Funktionseinheiten des PIC

4. Interruptsystem 8086

4.1. Interruptorganisation

Die CPU 8086 besitzt zwei Interrupteingänge für nichtmaskierbare (NMI) und maskierbare (INTR) Interrupts. Maskierbare Interrupts werden durch das CPU-Interrupt-Flag freigegeben oder gesperrt. Für 256 mögliche Interrupt-Service-Routinen sind die Startadressen mit Offset- und Segmentanteil in einer 1-KByte-Interrupt-Tabelle zu Beginn des Speicher-Adreßraumes nach Bild 4.1 plaziert. Die Interrupts werden mit dem Index der Adreßeintragung in der Interrupttabelle gekennzeichnet.

Die ersten fünf Interrupts sind nichtmaskierbar und CPU-internen Funktionen zugeordnet:

Interrupt Ø: von der CPU bei Division durch Null ausgelöst (divide error)

Interrupt 1: von der CPU nach jeder Befehlsausführung ausgelöst, falls T-Flag = 1 gesetzt ist (single step)

Interrupt 2: Nichtmaskierbarer Interrupt (NMI)

Interrupt 3: 1-Byte-Interrupt-Befehl INT3 (one byte interrupt)

Interrupt 4: von der CPU nach dem Befehl INTO ausgelöst, falls das Overflow-Flag gesetzt ist (on overflow)

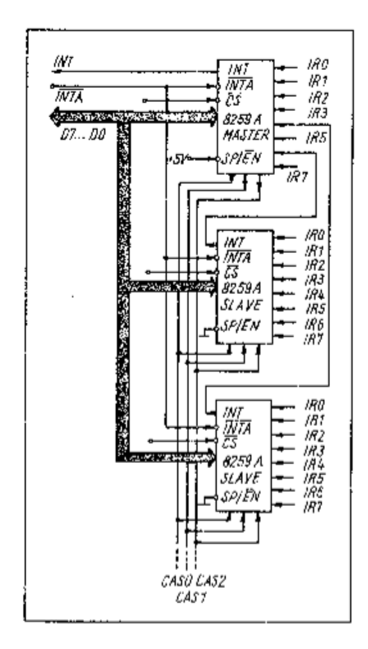
Für die Verbindung mehrerer Interruptquellen an den maskierbaren Interrupteingang INTR der CPU wird der programmierbare Interruptcontroller 8259A (PIC) eingesetzt, der speziell für das System 8086 entwickelt wurde (A-Typ erforderlich!).

Ein einzelner PIC kann 8 Interruptquellen verwalten und für diese die Prioritätsentscheidung übernehmen. Durch die Anschaltung von bis zu 8 Slave-PIC-Bausteinen an einen Master-PIC können maximal 64 unterschiedliche Interruptquellen im System verarbeitet werden.

Für eine über einen oder mehrere PICs ausgewählte Interruptanforderung wird der CPU-Eingang INTR auf '1' geschaltet. Die CPU reagiert nach dem Abschluß des in der Abarbeitung befindlichen Befehls mit zwei aufeinander folgenden Interruptbestätigungszyklen, die je einen INTA-Impuls an den PIC schalten (Bild 4.2).

Während des zweiten INTA-Impulses setzt der ausgewählte PIC einen der Interruptquelle zugeordneten Interruptvektor auf den niederwertigen Teil DBØ ... DB7 des Daten-BUS. Die CPU multipliziert diesen Vektor mit dem Wert 4. Damit entsteht ein Pointer auf die Adresse der Interrupt-Service-Routine in der Interrupttabelle. Die CPU übernimmt die Werte für das CS- und das IP-Register aus der Interrupttabelle, nachdem die Fortsetzungsadresse des unterbrochenen Programms mit Segment- und Offsetanteil im Stack abgespeichert worden ist. Zusätzlich wird im Stack das Flagregister, einschließlich des Interruptflags, gekellert. Das Interruptflag ist am Anfang der Interrupt-Service-Routine automatisch auf 'Ø' gesetzt worden.

Am Ende der Interrupt-Service-Routine werden die im Stack abgespeicherten Informa-



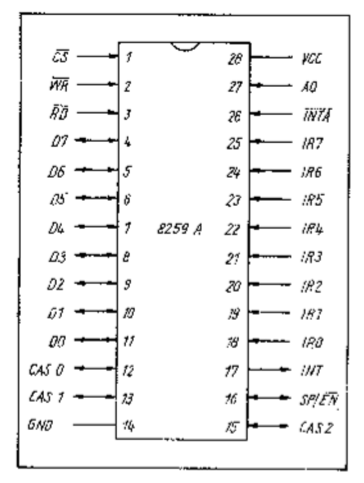


Bild 4.4 Kaskadierung von drei PICs

Bild 4.5 Pin-Belegung des 8259A

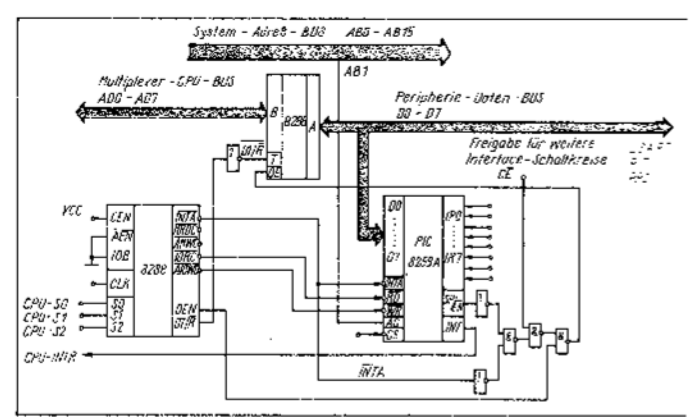


Bild 4.6 Anschaltung eines PIC in einen 8086-Mikrorechner (Maximum-Mode)

tionen mit dem Befehl IRET in das Flagregister, das IP- und CS-Register zurückgeschrieben. Damit ist der Zustand vor der Interruptannahme wieder hergestellt und mit dem Wert '1' für das Interruptflag eine erneute Interruptfreigabe der CPU gegeben.

Die Interrupt-Service-Routinen können auch mit Interruptbefehlen aufgerufen werden („Software-Interrupts"). Im 2-Byte-Interrupt-Befehl INT zz wird der zugehörige Interruptcode zz angegeben. Die Stackeintragung entspricht einem Hardware-Interrupt.

4.2. Aufbau und Funktionsprinzip des programmierbaren Interruptcontrollers 8259A

Nach Bild 4.3 enthält der PIC die Funktionseinheit für den System-BUS-Anschluß mit Daten-BUS-Puffer für den niederwertigen Teil DB0 ... DB7 des Daten-BUS und der Anschaltung der Steuersignale $\overline{RD}$ und $\overline{WR}$. Das Signal $\overline{CS}$ selektiert den Baustein innerhalb des E/A-Adreßraumes. Mit dem Eingang A0, der mit einer unteren Bitleitung des Adreß-BUS verbunden wird (z. B. AB1), werden die zwei möglichen Adressen für die Steuerkanäle des PIC unterschieden.

Die Prioritätsentscheidung für die 8 Eingangssignale an den Interrupt-Anforderungseingängen IR0 ... IR7 übernimmt eine Prioritätslogik im Zusammenwirken mit drei Registern.

Das Interruptrequestregister (IRR) speichert alle Interruptanforderungen an den Interruptrequesteingängen IR0 ... IR7. Die Interruptanforderung wird entweder mit der '0/1'-Flanke (edge triggered mode) oder mit dem '1'-Pegel (level triggered mode) übernommen. Die Ausgänge des IRR werden mit dem Interrupt-Masken-Register (IMR) maskiert.

Innerhalb der Interruptbestätigung durch die CPU wird die höchstpriorisierte Anforderung bei Berücksichtigung der Maskierungsbedingung aus dem IRR mit dem ersten INTA-Impuls in ein In-Service-Register (ISR) übernommen, falls in diesem keine höherpriorisierte Interruptabarbeitung markiert ist. Im ISR-Register sind also alle in der Abarbeitung befindlichen Interrupts notiert. Das höchste gesetzte Bit des ISR wird entweder am Ende der Interrupt-Service-Routine mit einem speziellen Kommando an den PIC oder automatisch nach dem zweiten INTA-Impuls (AEOI-Mode des PIC) gelöscht.

Die Kaskadierungslogik gestattet die Anschaltung von Slave-PICs an einen Master-PIC. Nach Bild 4.4 werden alle PICs an den Daten-BUS, die Steuersignale $\overline{RD}$ und $\overline{WR}$ und die Adreßauswahlsignale angeschlossen. Die Interruptausgänge INT der Slave-PICs sind mit den IR-Eingängen des Master-PIC verbunden.

Die Betriebsart jedes PIC als Master oder Slave wird durch die Programmierung und den Anschluß $\overline{SP}$ eingestellt. Der in einem Slave-PIC ausgewählte höchstpriorisierte Interrupt wird an den IR-Eingang des Master-PIC gelegt und von diesem nach den Prioritätsbedingungen der Masterebene an den INTR-Eingang der CPU geschaltet.

Die Interruptbestätigungszyklen werden von allen PICs parallel ausgewertet. Der Master-PIC legt auf die Ausgänge CAS0, CAS1 und CAS2 den Identifikationscode desjenigen Slave-PICs der den Interruptvektor auf den Daten-BUS zu geben hat.

4.3. Anschlußbeschreibung des 8259A

Die Pin-Belegung des 8259A zeigt Bild 4.5. Die Anschlüsse haben die folgende ***Bedeutung:***

$\overline{CS}$	Bausteinauswahl (chip select), Eingang
$\overline{WR}$	Schreibsignal (write), Eingang
$\overline{RD}$	Lesesignal (read), Eingang
D7–D0	Datenleitungen für das Schreiben von Steuerkommandos und das Lesen von Statusinformationen und des Interruptvektors, bidirektional, tri-state
CAS0, CAS1, CAS2	Kaskadierungssignale, Ausgänge für Master-PIC, Eingänge für Slave-PICs
$\overline{SP}$/$\overline{EN}$	Programmierung im Puffermode: $\overline{EN}$, Ausgang Ansteuersignal für Daten-BUS-Lesetreiber Programmierung im Nichtpuffermode: $\overline{SP}$, Master/Slave-Selektion, Eingang $\overline{SP}$ = 1: Master $\overline{SP}$ = 0: Slave
INT	Interruptausgang
IR0–IR7	Interruptanforderungseingänge, pegel- oder flankengesteuert (interrupt request)
$\overline{INTA}$	Interruptbestätigungssignal des Buscontrollers 8288 (interrupt acknowledge), Eingang
A0	Portauswahlsignal, Eingang, Verbindung mit AB1 des Adreß-BUS üblich)
Vcc	+5 V
GND	Masse

Die schaltungstechnische Einordnung des PIC 8259A im System 8086 zeigt Bild 4.6.

4.4. Programmierung des 8259A

Der 8259A ist sowohl für den Einsatz in 8-Bit-Prozessorsystemen (8080, 8085) als auch für die 16-Bit-Prozessoren 8086 und 8088 geeignet. Im folgenden wird die Programmierung nur für die 16-Bit-Prozessoren erläutert.

Die Programmierung des PIC erfordert zuerst eine Grundinitialisierung mit 2 bis 4 **Initialisierungskommandos**. Ein erstes Initialisierungskommandowort *ICW1*, ausgegeben auf Steuerport mit A0 = 0, startet die Programmierung. ICW1 enthält die Einstellung der Interruptrequesteingänge auf die Flanken- oder Pegelsteuerung und die Angabe, ob ein PIC oder mehrere kaskadierte PICs im System enthalten sind.

Das ebenfalls in allen Fällen notwendige zweite Initialisierungskommando *ICW2* enthält die 5 werthöchsten Bitstellen des Interruptvektors, der im zweiten Interruptbestätigungszyklus vom PIC auf den Daten-BUS gelegt wird. Der PIC setzt entsprechend der bestätigten Interruptanforderung die Bitpositionen T0, T1 und T2 des Interruptvektors ein.

wird fortgesetzt

Literatur:

/1/ Peripheral Design Handbook, Intel 1978

Bustreiberaufsatz D002

Detlef Poppe
VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen

Bisher waren die Einsatzmöglichkeiten der Module des KC-85/2-(/3-)Systems auf die beiden im Grundgerät befindlichen Modulschächte beschränkt. Der in diesem Beitrag vorgestellte Bustreiberaufsatz D002 vergrößert die Anzahl der verfügbaren Modulschächte um vier.
Es werden die technische Konzeption des Gerätes und durch die Erweiterung entstandene, beim Moduleinsatz zu beachtende Besonderheiten, wie WAIT-Erzeugung und Einstellung der Geräteadresse, genannt. Zu den beschriebenen technischen Details gehört auch die speziell beim KC-85-System verwendete Modulprioritätskette.
Zum Abschluß des Beitrages wird in einem Anwendungsbeispiel eine prinzipielle Lösungsmöglichkeit gezeigt, wie von einem KC 85/2 bzw. KC 85/3 Datenmengen verwaltet werden können, welche insgesamt den Adreßraum des im Computer eingesetzten Prozessors überschreiten.
Die Anwendungsmöglichkeiten des Bustreiberaufsatzes beschränken sich natürlich nicht auf die im Beispiel angegebene Modulkonfiguration und die Dateiverwaltung. Hierbei sind nur durch die technischen Möglichkeiten der Geräte und durch den Einfallsreichtum der Anwender Grenzen gesetzt.

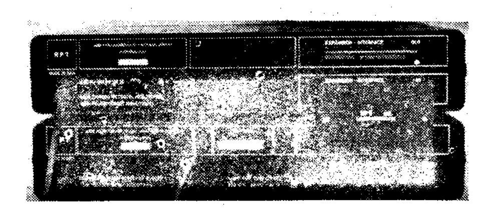
1. Einordnung des Gerätes in das Gesamtkonzept

Der Grund für die Entscheidung zugunsten eines KC 85/2 bzw. KC 85/3 liegt neben den grafischen Möglichkeiten auch in der Erweiterbarkeit des Gerätes. In der Gesamtkonzeption sind zwei Grundformen von Erweiterungsbausteinen vorgesehen, nämlich Module und Aufsatzgeräte. Bei den Modulen handelt es sich um steckbare Baugruppen, welche in die zwei am Grundgerät vorhandenen Modulschächte gesteckt werden können. Bekannte Beispiele hierfür sind der Modul M 006 (BASIC) und der Modul M 022 (EXPANDER RAM 16 KByte).
Bei den Aufsatzgeräten handelt es sich um solche Erweiterungen, die sich nicht in einem Modul unterbringen lassen. Gründe hierfür sind die Größe der unterzubringenden Funktionsgruppen, die Leistungsaufnahme oder die Tatsache, daß die Erweiterungen selbst Module aufnehmen sollen. Die Geräte befinden sich in einem in Abmessungen und Gestaltung auf das Grundgerät abgestimmten Gehäuse, welches auf das Grundgerät gestellt wird (Bild 1). Die Aufsatzgeräte enthalten ein eigenes Netzteil.
Die Verbindung der Signalleitungen mit dem Expansionsinterface des Grundgerätes oder eines anderen Aufsatzgerätes erfolgt mit Geräteverbindern (DEVICE CONNECTOR), welche im wesentlichen aus zwei miteinander verbundenen 58poligen Steckverbindern bestehen (Bild 2).
Der Aufsatz D002, welcher den Anschluß von vier zusätzlichen Modulen ermöglicht, wird neu in das Produktionsprogramm des VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen aufgenommen.



Bild 1 Beispiel einer Gerätekonfiguration mit Aufsatz D002 BUSDRIVER

Bild 2 Rückansicht der Geräte mit Geräteverbinder (Fotos: Mock)



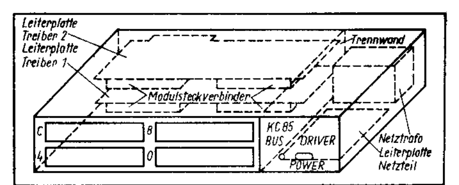
2. Hardware des Bustreiberaufsatzes und Hinweise zu seiner Anwendung

Der vom Grundgerät als Expansionsinterface herausgeführte Rechnerbus ist nicht getrieben, das heißt, die Belastungsmöglichkeit der Signalleitungen ist durch die Ausgangsbelastbarkeit der CPU begrenzt. Eine zusätzliche Belastung mit vier Modulen ist zu groß. Deshalb sind in den Signalleitungen zu den Modulschächten im D002 Treiberschaltkreise zwischengeschaltet, wovon sich auch der Name des Aufsatzgerätes ableitet. Der getriebene Rechnerbus wird über einen zweiten Expansionsinterfacestecker herausgeführt, um weitere Aufsatzgeräte anschließen zu können.
Bild 3 und 4 vermitteln einen Eindruck vom Aufbau des Gerätes. Im folgenden Abschnitt werden die Inbetriebnahme des Aufsatzes und einige Besonderheiten, die nicht unmittelbar aus den Abbildungen ersichtlich sind, erläutert.

2.1. Aufstellung und Inbetriebnahme

Der Aufsatz wird so auf das Grundgerät gestellt, daß die Füße des Gehäuses in die Aussparungen auf der Oberseite des Grundgerätes einrasten. Die elektrische Verbindung zum Expansionsinterface des Computers wird, wie bereits erwähnt, mit dem Geräteverbinder hergestellt. Außer dem Netzanschluß sind dann keine weiteren Verbindungen zum D002 erforderlich. Im Bedarfsfall kann noch ein weiteres Aufsatzgerät auf diesen gestellt und mit einem weiteren Geräteverbinder angeschlossen werden. Es ist empfehlenswert, die Aufsatzgeräte vor dem Grundgerät einzuschalten, weil durch das Betriebssystem die Module in einen definierten Anfangszustand rückgesetzt werden.

Bild 3 Konstruktiver Aufbau



Module dürfen nur im ausgeschalteten Zustand des Gerätes gesteckt bzw. gezogen werden.

2.2. Funktionsweise

Für den Datenbus werden bidirektionale Bustreiber (DS 8286) eingesetzt. Die Richtungssteuerung dieser Treiber wird von der aufsatzinternen Steuerelektronik realisiert (Bild 4). Für den Adreßbus und diejenigen Steuerleitungen, welche grundgeräteseitig Ausgänge darstellen, werden Treiberschaltkreise DL 541 bzw. Leistungsgatter eingesetzt. Für diejenige Gruppe von Steuerleitungen, welche Sammelleitungscharakter hat (ein Empfänger, mehrere Sender), hat der Bustreiber Open-Collector-Ausgänge (INT, NMI, WAIT usw.).
Die Signale BUSAK und BUSRQ werden im Bustreiberaufsatz nicht weitergeführt, weil im vorhandenen bzw. geplanten Modulsortiment kein DMA-Betrieb vorgesehen ist. Falls der Anwender über selbstgebaute Module mit DMA-Möglichkeit verfügt, können diese nur im Grundgerät betrieben werden.
Das beim Grundgerät realisierte Konzept der Modulsteuerung wird beim Bustreiberaufsatz konsequent weitergeführt. Das Schalten der Module und die Strukturbyteabfrage erfolgen über die I/O-Adresse 80H (siehe /1/).
Auf dem höherwertigen Teil des Adreßbus wird eine schachtspezifische Adresse ausgegeben, deren höherwertige Tetrade (Geräteadresse) für die Geräteauswahl zuständig ist, während die niederwertige Tetrade die Auswahl des Schachtes innerhalb des Gerätes bewirkt.

2.3. Moduladressierung

Die Geräteadresse ist beim D002 im Regelfall auf 1 eingestellt. Die Schachtadressen lauten also 10H, 14H, 18H und 1CH. Falls zwei Bustreiberaufsätze an einem Grundgerät betrieben werden sollen, muß die Geräteadresse des zweiten Aufsatzes verändert werden, beispielsweise in 2. Das ist im Bereich von 1 bis F durch im Gerät befindliche Lötbrücken möglich. Die Änderung der Lötbrücken wird von den Servicewerkstätten vorgenommen.
Im Grundgerät bzw. Aufsatz wird aus der Adresse für jeden Schacht ein spezifisches Selektionssignal ausdekodiert (/MAD8 für Schacht 08, /MADC für Schacht 0C, /MAD10 usw.) – siehe Bild 5a.
Auf diesem Bild ist weiterhin die Modul-Prioritätskette dargestellt. Ihre Funktion besteht darin, Zugriffskonflikte auf dem Datenbus zu vermeiden. Die Wirkungsweise ist mit der bekannten Interruptprioritätskette vergleichbar.
Nehmen wir an, daß in allen Schächten des Bustreiberaufsatzes RAM-Module gesteckt sind, welche auf die Basisadresse 4000H geschaltet sind. Die Module 14, 18 und 1C seien aktiv geschaltet. Dann würde beispielsweise der Befehl LD A, (4010H) die Speicherschaltkreise in allen drei aktiven Modulen gleichzeitig zur Datenausgabe veranlassen. In diesem Fall gäbe es auf dem Datenbus nicht immer eindeutig als LOW oder HIGH bewertbare Pegel. Das wird aber durch die Prioritätskette verhindert. Der Modul kann nur dann Daten ausgeben, wenn er aktiv geschaltet ist und an seinem MEI-Eingang HIGH-Pegel anliegt. Wenn er Daten ausgibt, gibt der Modul an seinem eigenen MEO-Ausgang LOW-Pegel aus. Ein nicht angesprochener Modul gibt den an seinem Eingang anliegenden HIGH-

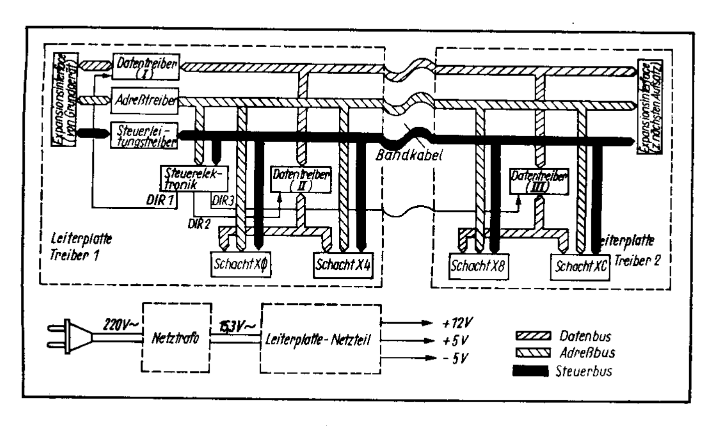


Bild 4 Blockschaltbild

Bild 5 Modulprioritätskette
a) Prinzip der Prioritätskette. Die angegebenen Pegel (H,L) beziehen sich auf den Fall, daß auf den Modul mit der Adresse 14 zurückgegriffen wird.
b) Umgehungslogik
c) Mögliche Bestückungsvarianten

Pegel an den Ausgang weiter. LOW-Pegel wird von allen Modulen weitergeleitet, unabhängig davon, ob sie aktiv geschaltet sind oder nicht.
In unserem Beispiel gelangt der HIGH-Pegel der Prioritätskette bis zum Schacht 14. Dieser Modul kann auf dem Datenbus seine Daten ausgeben und gibt auf der Prioritätskette LOW-Pegel weiter. Damit wird die Datenausgabe der Module 18 und 1C verhindert. Von der beim U880-System allgemein verwendeten Interruptprioritätskette sind verschiedene Lösungen zur Verkürzung der Signallaufzeiten mittels Umgehungsgattern bekannt. Die im D002 für Interrupt- und Speicherprioritätskette verwendete Variante wird in Bild 5b gezeigt. Die Ein- und Ausgangssignale der Prioritätskette haben nicht nur die beschriebene Funktion, sie werden auch von der Richtungssteuerung der Datentreiber benutzt.

2.4. Modulbestückung

Bei Bestückung des Aufsatzes mit einer geringeren Anzahl als der maximal möglichen von vier Modulen sind bestimmte Bedingungen zu beachten, damit beide Prioritätsketten ordnungsgemäß funktionieren können. Wird eine Modulebene (d. h. zwei nebeneinanderliegende Schächte) mit nur einem Modul bestückt, muß dieser in den rechten Schacht gesteckt werden. Wird er in den linken Schacht gesteckt, kann es zu Komplikationen kommen, was man sich leicht am Bild 5b klarmachen kann. Wenn in diesem Bild Modul 14 gesteckt und Modul 10 nicht gesteckt ist und vom Ausgang des Moduls 08 LOW-Pegel gesendet wird, wird dieser zwar über das AND-Gatter an die nachfolgenden Schächte weitergeleitet, der Modul 14 hat aber an seinem Eingang MEI HIGH-Pegel. In diesem Fall ist ein Datenbuskonflikt zwischen Modul 14 und dem Grundgerät bzw. einem im Grundgerät gesteckten Modul möglich. Die drei zulässigen Bestückungsvarianten einer Modulebene (siehe Bild 5c) können in den beiden Modulebenen des Aufsatzes beliebig kombiniert werden.
Im Grundgerät fehlen die in Bild 5b angegebenen Widerstände. Zur störungsfreien Funktion der Gerätekonfiguration ist es deshalb erforderlich, daß bei der Nutzung von Aufsatzgeräten beide Schächte des Grundgerätes mit Modulen bestückt sind.

2.5. WAIT-Erzeugung

Der D002 besitzt eine interne WAIT-Logik. Im Normalzustand des Aufsatzes wird bei jedem M1-Zugriff auf einen im Aufsatz steckenden Modul oder einen nachfolgenden Aufsatz ein WAIT mit der Dauer einer Taktperiode eingeblendet.
Diese Maßnahme wurde mit Rücksicht auf die insbesondere bei der Arbeit mit mehreren Aufsatzgeräten wirksam werdende Verlängerung der Signalleitungen und die Signallaufzeiten der Modulprioritätskette vorgesehen, sie ist aber nicht in jedem Fall erforderlich. Aufgrund bisher vorliegender Erfahrungen hat sich der in einigen Modultypen eingesetzte EPROM U 2716 im Verhältnis zu anderen Speichertypen als empfindlicher gegenüber Leitungsstörungen erwiesen.
Damit im Bedarfsfall das zeitliche Verhalten optimiert werden kann, ist die Möglichkeit vorgesehen, die Funktion der WAIT-Logik mittels Lötbrücken zu verändern. Das aufsatzintern erzeugte WAIT kann total unterdrückt bzw. auf die anderen Arten von Prozessorzugriffen erweitert werden.
Wenn der Modul M 027 DEVELOPMENT in den Aufsatz gesteckt wird, ist zu beachten, daß, bedingt durch das WAIT, die Schrittbetriebsart des Testmonitors nicht funktioniert. Die anderen Funktionen des Moduls werden aber nicht beeinträchtigt.
Wenn die Schrittbetriebsart genutzt werden soll, ist der Modul in das Grundgerät zu stekken, bzw. das aufsatzinterne WAIT muß abgeschaltet werden.
Die eben erwähnte WAIT-Problematik ist auch bei Programmabläufen zu berücksichtigen. Beispiele hierfür sind Zeitschleifen, welche durch Dekrementieren von Registern realisiert werden und Programme mit Echtzeitforderungen, wie es beispielsweise bei Antriebssteuerungen der Fall ist. Es gibt in solchen Fällen Unterschiede der Programmlaufzeit, je nachdem, ob der Modul mit dem Programm im Grundgerät oder im Aufsatz steckt.

2.6. Modulverwaltung

Das Schalten und die Strukturbyteabfrage des Moduls erfolgen in der vom Grundgerät bekannten Art und Weise. Neben der Möglichkeit, auf Maschinensprachebene über geeignete IN- bzw. OUT-Befehle direkt auf die I/O-Adresse 80H zuzugreifen, bieten Betriebssystem, FORTH und BASIC-Interpreter komfortablere Möglichkeiten in Form des in der Unterprogrammtabelle enthaltenen Unterprogramms MODU mit der UP-Nr. 26H (siehe /3/) bzw. in FORTH mit den Worten MODUL und SWITCH /4/ sowie in BASIC mit der Anweisung SWITCH mm, kk (siehe /2/). Die Modulsteuerung kann auch mit den Funktionen SWITCH und JUMP aus dem CAOS-Menü angesprochen werden.
Beim Stecken der Module ist unter Umständen auf eine zweckmäßige Reihenfolge in der Prioritätskette zu achten. So müssen beispielsweise 64-KByte-RAM-Module an den letzten Positionen der Kette eingeordnet werden, weil sie im aktiv geschalteten Zustand den Zugriff auf alle dahinter liegenden Speichermodule blockieren würden.

3. Anwendung des D002

Die Möglichkeiten des Kleincomputersystems KC 85/2(/3) konnten bisher aufgrund der durch das Grundgerät und dessen zwei Steckplätze beschränkten Speicherkapazität nicht voll ausgenutzt werden, weil viele Anwendungsfälle große Datenmengen erfordern, die ohne zusätzliche Ladevorgänge schnell verfügbar sein müssen. Beispiele hierfür sind Buchhaltung, Lagerverwaltung und ähnliches. Zur Lösung des Problems gibt es zwei grundsätzliche technische Möglichkeiten, die aber auch kombiniert angewendet werden können.
Die eine Möglichkeit besteht in gegenüber der Magnetbandkassette verbesserten magnetomotorischen Speichermedien (Diskette, Festplatte).
Der andere Weg besteht in einer wesentlichen Vergrößerung der RAM-Kapazität gegenüber der Grundausstattung des KC-Systems. Diese Möglichkeit, die im folgenden näher beschrieben wird, kann unter Verwendung des Bustreiberaufsatzes D002 realisiert werden, der für diesen Zweck mit vier Stück M 022 EXPANDER RAM oder noch besser M 011 64 KByte RAM bestückt wird (in den folgenden Ausführungen wird der allgemeine Oberbegriff RAM-Modul verwendet). Ein Vorteil dieser Lösung gegenüber magnetomotorischen Speichern liegt in der kürzeren Zugriffszeit.
Die Daten müssen aber einmalig vor Beginn der Verarbeitung vom magnetomotorischen Speicher (Kassette) geladen werden und am Ende der Bearbeitung bzw. nach wichtigen Zwischenstufen auf diesen gerettet werden. Beim Speichermedium Kassette sind diese Vorgänge zwar mit einem gewissen Zeitaufwand verbunden; wenn ein leistungsfähigeres Massenspeichermedium nicht zur Verfügung steht, ist diese Vorgehensweise aber für viele Anwendungsfälle zweckmäßig. Für die oben genannten Anwendungsfälle müssen häufig Ergebnisse ausgedruckt werden. In Abhängigkeit von der geforderten Druckerschnittstelle muß ein entsprechender Modul-

typ ausgewählt werden, der im Grundgerät gesteckt wird. Meistens kommt ein M003 (V.24) in Frage. Der zweite Schacht des Grundgerätes kann in der Entwicklungs- und Testphase der Software mit einem M027 DEVELOPMENT belegt werden. Die Endfassung des Programms kann auf einem M025 USER PROM 8K gespeichert werden, der dann statt des Entwicklungssystems gesteckt wird.
Wird das Anwenderprogramm nicht in Maschinensprache, sondern in einer höheren Programmiersprache geschrieben, können auch ein FORTH-Modul (M026) bzw. Modul M006 BASIC (beim KC 85/2) verwendet werden. Bei Nichtinanspruchnahme dieser Möglichkeiten könnten auch weitere RAM-Module gesteckt werden.
Ein wesentliches Problem bei der Dateiarbeit liegt in der zweckmäßigsten Speicheraufteilung. Der Gesamtdatenbestand wird in Teilblöcke von je 16 KByte untergliedert, wobei je nach RAM-Modultyp 4 bzw. 16 Blöcke verfügbar sind. Für die Daten wird der Adreßbereich von 4000H bis 7FFFH verwendet, wobei je nach Bedarf der gerade benötigte Datenblock auf diesen Adreßbereich aktiv geschaltet wird. Für das Programm selbst, dessen Arbeitszellen usw. wird der Adreßbereich unterhalb 4000H verwendet. (Bei einem BASIC-Programm MEMORY END auf ≦ 16383 begrenzen!) Bei einer Realisierung des Anwenderprogramms als Maschinenprogramm auf EPROM wird dieses in den Adreßbereich 0C000H bis 0DFFFH gelegt. (Beim KC 85/3 BASIC-ROM mit SWITCH 2 0 abschalten!)
Die Zu- und Abschaltung der jeweils benötigten Datenblöcke muß vom Anwenderprogramm selbst organisiert werden. Hierzu steht der bereits genannte Befehl SWITCH zur Verfügung.
Eine Realisierung des Anwenderprogramms in BASIC ist möglich, aber für umfangreiche Datenbestände im allgemeinen zu langsam. Ein Vorteil des BASIC-Programms liegt jedoch in der Entwicklungsphase. In BASIC kann selbst der in der Programmierung relativ unerfahrene Anwender Änderungen der Verarbeitungsalgorithmen oder Bildschirmdarstellungsformate leicht ausprobieren.
Der Zugriff auf die außerhalb des eigentlichen BASIC-Speicherbereichs liegenden Datenbestände erfolgt mittels POKE/PEEK bzw. DOKE/DEEK. Die Magnetbandspeicherung der Daten erfolgt in gleicher Weise wie bei Maschinenprogrammen, also mit SAVE, LOAD bzw. BLOAD.
Wenn das Programm im wesentlichen den Wünschen des Anwenders entspricht, kann der Nutzer mit Kenntnissen der Maschinensprache damit beginnen, zeitaufwendige Programmteile, wie beispielsweise Such- und Sortieralgorithmen, in Maschinenunterprogramme (Aufruf mit CALL) umzusetzen. Im Bedarfsfall kann schrittweise das gesamte BASIC-Programm durch Maschinenprogramme ersetzt werden. Die bereits vom BASIC-Programm gespeicherten Daten können vom Maschinenprogramm unverändert weiter benutzt werden.
Beim Laden und Retten der Inhalte der einzelnen RAM-Module mittels LOAD bzw. SAVE können die jeweiligen Datenblöcke mit der Menüfunktion SWITCH zugeschaltet werden. Hierbei kann man sich durch Funktionstastenbelegungen oder ein kleines Hilfsprogramm Erleichterungen schaffen.

Literatur

/1/ Kleincomputer KC 85/3 (System-Handbuch). VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen 1986
/2/ Kleincomputer KC 85/3 (BASIC-Handbuch). VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen 1986
/3/ Kleincomputer KC 85/3 (Übersichten). VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen 1986
/4/ Kleincomputer KC 85/3 M 026 FORTH. VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen 1987

✉ KONTAKT ☏

VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen, Eisenacher Straße 40, Mühlhausen, 5700; Tel. 53282 (Autor) oder 53209 (Applikationsstelle)

zur 2. Umschlagseite

Verhalten von Zahlenfolgen grafisch dargestellt

Detlef Thielsch, Wolfen

Zahlenfolgen sind häufig auf ihr Konvergenzverhalten zu untersuchen. Diese Untersuchung wird durch eine grafische Darstellung besonders anschaulich. Die in der Literatur /1/, /2/ bekannt gewordene Mandelbrotmenge (Benoit B. Mandelbrot – Mitarbeiter am IBM Forschungszentrum in Yorktown Heights) liefert darüber hinaus auch ästhetisch ansprechende Bilder. Sie stellt den Konvergenzbereich der wie folgt definierten Folge komplexer Zahlen dar:
Die Zahl $z_i = x + jy$ wird quadriert und die Konstante $c = x_c + jy_c$ addiert bzw. subtrahiert. Auf die so erhaltene Zahl z_{i+1} wird das Verfahren erneut angewandt:

$$z_{i+1} = z_i^2 \pm c \text{ mit } z_i^2 = x^2 - y^2 + j2xy$$
$$\text{und } z_0 = 0; j = 0, 1, 2, 3, \ldots$$

In Abhängigkeit von c ist die Folge konvergent oder divergent.
Untersucht man mit dem untenstehenden BASIC-Programm die Punkte der Gaußschen Zahlenebene in der Nähe des Koordinatenursprungs und setzt an der entsprechenden Stelle des Bildschirms für Konvergenz Schwarz, für Divergenz in Abhängigkeit von der Schnelligkeit des Anwachsens der Zahl z unterschiedliche Farbwerte, so resultiert Bild 1, das auch aus MP 1/88 bekannte „Apfelmännchen" (die Bilder zu diesem Beitrag finden Sie auf der 2. Umschlagseite).
Weniger geläufig sind die Ergebnisse der analogen Vorgehensweise für andere Potenzen von z, wie sie die Bilder 2 und 3 zeigen.

Für z^3 ist das Programm wie folgt zu ändern:
230 YZ = (XX + XX + XX − YY) × YZ − YC
240 XZ = (XX − YY − YY − YY) × XZ − XC
Für z^4:
230 YZ = (YZ + YZ + YZ + YZ) × (XZ × XX − (XZ × YY)) − YC
240 XZ = (XX × XX) − (6 × XX × YY) + (YY × YY) − XC

Weitere interessante Ergebnisse liefern Manipulationen am Programm. Beginnt die Iteration mit $z_0 = c$ und wird die zu c konjugiert komplexe Zahl $c = x_c - jy_c$ subtrahiert, so resultiert eine Birnenform

```
0 REM FRACTALS Z=Z^2+C , ATARI-BASIC
10 NN=109:MM=159:A=1:B=5:D=0
20 DIM F(5)
30 FOR I=1 TO 5:REM FARBZUWEISUNG
40 READ F:F(I)=F
50 NEXT I
60 DATA 3,2,1,3,2
70 TRAP 70:PRINT "ITERATIONSGRENZE(50-->1000] ";:INPUT KX
80 PRINT "EINGABE ABBRUCHSCHRANKE(100-->5000] ";:INPUT S
90 PRINT "ZU UNTERSUCHENDES INTERVALL"
100 TRAP 100:? "XU";:INPUT XU:? "XO";:INPUT XO
105 TRAP 105:? "YU";:INPUT YU:? "YO";:INPUT YO
110 DX=(XO-XU)/(NN+1)
120 DY=(YO-YU)/(MM+1)
130 GRAPHICS 15:REM 160*160 PUNKTE
140 YC=YU
150 FOR M=D TO MM
160 XC=XU
170 FOR N=D TO NN
180 XZ=D:YZ=D
210 FOR K=D TO KX
220 XX=XZ*XZ:YY=YZ*YZ
230 YZ=(XZ+XZ)*YZ-YC
240 XZ=XX-YY-XC
260 IF XX+YY>S THEN COLOR F(K-INT(K/B)*B+A):PLOT N,MM-M:K=KX+A
270 NEXT K
280 XC=XC+DX
290 NEXT N
300 YC=YC+DY
310 NEXT M
320 END
330 REM ***********************************************************
340 REM INTERVALLVORSCHLAG: X=[-1.7;2.3] Y=[-1.25;1.25]
350 REM GERINGERE LAUFZEIT DURCH 2.ZEICHENBEFEHL IN 260 (Y-SYM.)
360 REM AUSSCHNITTE MIT GROESSEREM S UND KX
370 REM ***********************************************************
```

(Bild 4). Werden für z^4 die Programmzeilen 230 und 240 vertauscht, so ergibt sich Bild 5. Die mehrstündigen Rechenzeiten können durch die Symmetrie zur realen Achse verkürzt werden. Bemerkenswert sind Ausschnittsvergrößerungen der Mengenränder. (Bild 6).

Literatur

/1/ Jordan, M.: Mandelbrotmenge schnell berechnen. MC (1987) 6, S. 120–121
/2/ Bizarr und zerbrechlich. HC (1986) 4, S. 100–103
/3/ Lautenbacher, M.: Apfelmännchen in PASCAL. MC (1986) 12, S. 76

Diagrammdarstellung auf dem PC 1715

Arnd Hilbert, Magdeburg

Innerhalb des Datenbanksystems REDABAS können numerische Werte in der Datei direkt gespeichert werden.
Zusätzlich lassen sich durch die Befehle SUM oder TOTAL ON Zahlenwerte ermitteln, die eine quantitative Einschätzung bestimmter Dateiinhalte ermöglichen.
Die grafische Darstellung dieser Werte in einem Säulen- oder Balkendiagramm eignet sich besonders für vergleichende Betrachtungen oder zur Illustrierung des Entwicklungstrends über einen bestimmten Zeitraum. Unter Verwendung der Terminalsteuerzeichen des PC 1715 sind Diagrammdarstellungen auch für die Programmiersprache des Datenbanksystems REDABAS einfach zu realisieren. Im folgenden soll ein Programmbeispiel zur Kostenanalyse (Kosten pro Monat) erläutert werden.
Entsprechende Zahlenwerte können im Programm als freie Speicherzellen oder als Felder einer Datei übergeben werden.

Einfacher geht's nicht mehr

Zur Dateneingabe werden in diesem Beispiel 12 Speicherzellen, bezeichnet mit den Buchstaben A bis L, deklariert. Da die numerischen Inhalte dieser Speicherzellen im Programm häufig nacheinander abgefragt werden, ist es sinnvoll, dieses durch eine entsprechende DO WHILE-Schleife zu realisieren.
Beginnend mit der Dezimalzahl 65 wird die Bezeichnung jeder einzelnen Speicherzelle durch die Charakter-Funktion CHR (⟨Zahl⟩) dargestellt: CHR(65)=A. Anstelle der direkten Zahleneingabe läßt sich eine weitere Speichervariable (Anfangswert: 65) einsetzen, deren Wert bei jedem Schleifendurchlauf um eins erhöht wird, bis die jeweilige Operation für alle Ausgangswerte durchgeführt wurde.
Eine wesentliche Vereinfachung des Programms ist die Folge.
Das gleiche Verfahren kann auch auf entsprechend bezeichnete Felder einer speziellen Datei angewandt werden. Die Datensätze einer solchen Datei ermöglichen außerdem eine weitere Sortierung der Daten (z. B. nach Jahren, Bereichen usw.).

Damit alles im Rahmen bleibt

Für eine Bildschirmbreite von 80 Zeichen läßt sich ein Säulendiagramm mit maximal 40 Werten erstellen (Reliefdiagramm: 80). Die Anzahl der Bildschirmzeilen (max. 24 Zeilen) bestimmt in diesem Fall die Höhe der Säule für den größten Wert. Ausgehend vom größten Ausgangswert wurde zur Darstellung der übrigen Werte ein Vielfaches von 1/20 des Maximalwertes gewählt.

Programmlistings

```
*                   Hauptprogramm DIAGRAMM
*                   ~~~~~~~~~~~~~~~~~~~~~~
set colon on
set talk off
*
* Initialisierung der Steuerzeichen
*
store 'CHR(27)+CHR(94)+CHR(80)' to INVERS
store 'CHR(27)+CHR(94)+CHR(64)' to NORM
store 'CHR(27)+CHR(94)+CHR(96)' to LINE
store 'CHR(27)+CHR(116)+CHR(49)' to GRAPHIK
store CHR(27) to ESC
*
* Darstellung des Koordinatensystems
*
store '1' to PS,NEU
do while PS='1'
         erase
         store 0 to SP,ZE
         @ 0,00 say '^'
         @ 0,32 say 'KOSTENANALYSE              Stand:'+DATE()
         do while ZE<21
                  @ (ZE+1),SP say '¦'
                  store (ZE+1) to ZE
         enddo
         @ 21,00 say &LINE+'                                    ;
                                                  >'+&NORM
*
* Dateneingabe
*
         if NEU='1'
            store 0 to A,B,C,D,E,F,G,H,I,J,K,L
         endif
         store 65 to X
         @ 22,03 say 'JAN   FEB   MAE   APR   MAI   JUN '
         @ 22,37 say '  JUL   AUG   SEP   OKT   NOV   DEZ'
         if NEU='1'
            do while X<77
                     store CHR(X) to Y
                     @ 23,SP say '  ' get &Y picture '###'
                     store (X+1) to X
                     store (SP+6) to SP
            enddo
            read
*
* Ermitteln des Maximalwertes &Y
*
            store CHR(75) to Z
            do while X>64
                     if (&Y-&Z)<0
                        store Z to Y
                     endif
                     store (X-1) to X
                     store CHR(X) to Z
            enddo
         endif
*
```

```
* Darstellung des Saeulendiagramms
*
         store 65 to X
         store (&Y/20) to Q
         store 3 to SP
         @ ZE,SP say CHR(131)
         do while X<77
                  store CHR(X) to Z
                  store (&Z/Q) to DIF
                  store (20-INT(DIF)) to STOP
                  store 20 to ZE
                  do while ZE<>STOP
                           @ ZE,SP say &INVERS+'  '+&NORM
                           store (ZE-1) to ZE
                  enddo
                  store (SP+6) to SP
                  store (X+1) to X
         enddo
*
* Unterprogrammaufruf
*
         store '?' to UP
         do while UP<>'ENDE'
                  store 65 to X
                  do case
                          case UP='0'
                               store 'ENDE' to UP,PS
                          case UP='1'
                               store '1' to NEU,PS
                               store 'ENDE' to UP
                          case UP='2'
                               do BALKEN
                          case UP='3'
                               do DRUCK1
                          case UP='4'
                               do WERTE
                   otherwise
           @ 23,00 say 'Bitte Kennzahl eingeben:Beenden-0 Neu-1 Balken'
           @ 23,46 say 'diagramm-2 Druck-3 Werte-4' get UP picture 'X'
                   read
                   endcase
          enddo
enddo
erase
@ 0,0 say CHR(130)
return
*
```

```
*                   Unterprogramm BALKEN
*                   ~~~~~~~~~~~~~~~~~~~~
* Darstellung des Balkendiagramms
*
* Koordinatensystem
*
store 'JAN FEB MAE APR MAI JUN JUL AUG SEP OKT NOV DEZ ' to W
erase
@ 23,05 say &LINE+'                                        ;
                  >' +&NORM+'KOSTEN'
*
* Balkendarstellung:
*
store '                                                    ;
       ' to U
store 45 to V
store 22 to ZE
store 76 to X
store (&Y/62) to Q
do while X>64
         store CHR(X) to Z
         store INT(&Z/Q) to LAENGE
         if LAENGE>0
            store ¤(U,1,LAENGE) to ROW
            @ ZE,0 say ¤(W,V,4)+' ¦'+&INVERS+ROW+&NORM+&NORM
          else
               @ ZE,0 say ¤(W,V,4)+' ¦'
         endif
         @ (ZE-1),05 say '¦'
         store (X-1) to X
         store (V-4) to V
         store (ZE-2) to ZE
enddo
*
* Programm fortsetzen
*
store '?' to PS
@ 0,71 say '*S.diag-1'
@ 1,71 say '*Druck -5'
@ 2,71 say 'Ende-0' get PS Picture 'X'
read
do while PS='5'
         do DRUCK2
enddo
store '0' to NEU
store 'ENDE' to UP
return
*
```

```
*               Unterprogramm DRUCK1
*
* Ausdrucken des Saeulendiagramms
*
* Ueberschrift
*
set format to print
@ 0,30 say ESC+'W1'+ESC+'-1'+'KOSTENANALYSE'+ESC+'WO'+ESC+'-0'
@ 0,60 say 'Stand:'+DATE()+&GRAPHIK
@ 1,05 say '^'
*
* Schleife 1:Ausdrucken der Zeilen 1-20
*
store 2 to ZE
do while ZE<22
         store 65 to X
         store 7 to POS
         @ ZE,05 say CHR(186)
*
* Schleife 2:Ausdrucken einer Zeile
*
         do while X<77
                   store CHR(X) to Z
                   store (&Z/Q) to DIF
                   store INT(DIF) to STOP
                   if STOP>(20-ZE)
                      @ ZE,POS say CHR(219)+CHR(219)+CHR(219)
                   endif
                   store (POS+6) to POS
                   store (X+1) to X
         enddo
         store (ZE+1) to ZE
enddo
*
* Ausdrucken der X-Achse und der Monatsnamen
*
store 6 to POS
@ ZE,05 say CHR(212)
do while POS<77
         @ ZE,POS say CHR(205)
         store (POS+1) to POS
enddo
@ ZE,77 say '>'
@ (ZE+1),07 say 'JAN   FEB   MAE   APR   MAI;
  JUN   JUL   AUG   SEP   OKT   NOV   DEZ'
*
* Werte ausdrucken
*
store 7 to POS
store 65 to X
do while X<77
         store CHR(X) to V
         @ (ZE+2),POS say STR(&V,3,0)
         store (X+1) to X
         store (POS+6) to POS
enddo
set format to screen
store '?' to UP
return
*
```

```
*               Unterprogramm WERTE
*
* Einblenden der Werte
*
store 0 to POS
do while X<77
         store CHR(X) to V
         @ 23,POS say '    '+STR(&V,3,0)
         store (X+1) to X
         store (POS+6) to POS
enddo
store '?' to PS
@ 23,72 say '      ' get UP picture 'X'
read
return
*
```

```
*
*               Unterprogramm DRUCK2
*
* Ausdrucken des Balkendiagramms
* Ueberschrift:
set format to print
@ 0,30 say ESC+'W1'+ESC+'-1'+'KOSTENANALYSE'+ESC+'WO'+ESC+'-0'
@ 0,60 say 'Stand:'+DATE()+&GRAPHIK
*
* Schleife 1: Balkendarstellung
store 65 to X
store 1 to V
store 2 to ZE
do while X<77
         store CHR(X) to Z
         @ ZE,07 say ¤(W,V,3)+':'+STR(&Z,3,0)+CHR(222)
         store 15 to POS
         store INT(&Z/Q) to LAENGE
*
* Schleife 2: Ausdrucken eines Balkens
         do while (POS-15)<LAENGE
                   @ ZE,POS say CHR(219)
                   store (POS+1) to POS
         enddo
         @ (ZE+1),14 say CHR(222)
         store (ZE+2) to ZE
         store (X+1) to X
         store (V+4) to V
enddo
*
* Druck der X-Achse
store 14 to POS
do while POS<70
         @ ZE,POS say CHR(196)
         store (POS+1) to POS
enddo
@ ZE,POS say '>'+'KOSTEN'
*
* Programm fortsetzen
*
set format to screen
store '?' to PS
@ 2,76 say '0' get PS picture 'X'
read
return
*
```

Die Höhe einer Säule zeigt damit die Abweichung vom größten Wert in der entsprechenden Abstufung mit einer Genauigkeit von 1/20 dieses Wertes.
Eine höhere Auflösung ist bei der zeilenweisen Darstellung aufgrund der geringen Bildschirmhöhe nicht möglich.

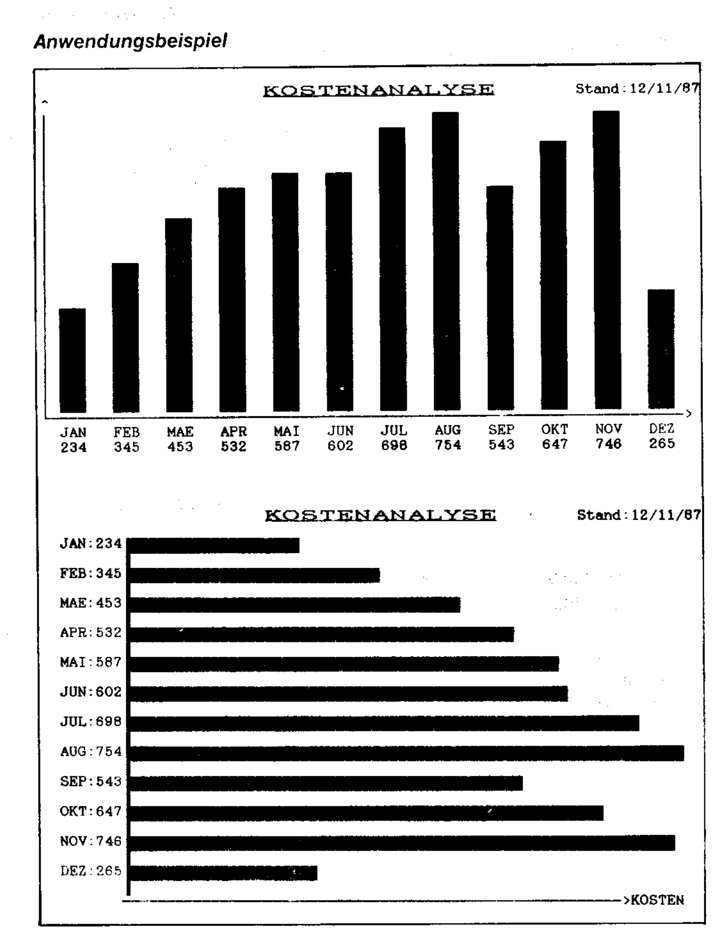
Was man Schwarz auf Weiß besitzt...

Die Ausgabe des Säulendiagramms auf den Drucker gestattet einen besseren Vergleich der Werte, da in Abhängigkeit von der Blatthöhe die Genauigkeit bis auf 1/60 des Maximalwertes gesteigert werden kann. Über eine Kommentarzeile auf dem Bildschirm kann der Nutzer ein spezielles Druckprogramm (DRUCK1.CMD) aufrufen, das den Ausdruck des dargestellten Diagramms auf einem Nadeldrucker (EPSON LX-86) realisiert.
Mit einem Balkendiagramm kann die Abbildung ebenfalls mit größerer Genauigkeit erfolgen, da die Anzahl der Bildelemente über die gesamte Bildschirmbreite maximal 80 Abstufungen zuläßt.
Allerdings reduziert sich hierbei die Anzahl der maximal darstellbaren Werte auf 24, da in jeder Zeile nur ein Balken abgebildet werden kann. Das vorliegende Beispielprogramm bietet die Möglichkeit des Vergleichs beider Darstellungsarten. Über die Eingabe der entsprechenden Kennzahl können Unterprogramme zur Darstellung des Balkendiagramms (BALKEN.CMD) oder zum Einblenden der aktuellen Zahlenwerte (WERTE.CMD) gestartet werden.
Das Druckprogramm DRUCK2.CMD gestattet die Ausgabe des Balkendiagramms auf dem Drucker.

Literatur

/1/ Matzke, B.: Terminalsteuerzeichen des PC 1715. Mikroprozessortechnik 1 (1987) 5, S. 140–141.

Anwendungsbeispiel



Die Arbeit mit Direktzugriffsdateien

Michael Lennartz, Ludwigsfelde

Disketten und Festplatten als Bestandteil von Mikrorechnern bestimmen wesentlich das Leistungsniveau dieser Geräte. Sie sind gleichzeitig typische Datenträger für die Arbeit mit Direktzugriffsdateien (DA-Dateien). Viele BASIC-Dialekte wie auch die BASIC-80-Systeme BASI (Interpreter) und BASC (Compiler) von Robotron beinhalten leistungsfähige Anweisungen zur Arbeit mit DA-Dateien. Diese sollen im folgenden erläutert werden.

1. Eröffnen von Dateien

In BASIC werden Nummern zum Ansprechen von peripheren Geräten verwendet. Um eine Beziehung zwischen der intern zugewiesenen Nummer und der physischen Diskettendatei herzustellen, wird mit der OPEN-Anweisung die Datei eröffnet:

OPEN "R",#n,name,length

Es bedeuten:

"R" – Kennzeichen für Direktzugriff

#n – Zahl, unter der die Datei programmintern geführt wird. Das Zeichen # kann entfallen. Für n gilt: 1<=n<=15.

name – Zeichenkette, die den Namen der Datei angibt. name ist eine Zeichenkettenvariable oder wird in Hochkomma eingeschlossen.

length – Länge des Ein-/Ausgabepuffers, der vom System bereitgestellt werden muß.

In der Interpreterversion ist n auf maximal 3 und length auf maximal 128 eingestellt. Bei der Arbeit mit dem Interpreter können diese Werte durch den Aufruf

basi /f:n /s:laenge

auf die Verarbeitung von höchstens 15 Dateien und eine größere Satzlänge eingestellt werden.

Beispiel:

OPEN "R",#1,"NAME.FIL",20 bzw.
OPEN "R",#1,"B:NAME.FIL",20
N○ = "NAME.FIL"
LW○ ="B:"
OPEN "R",#1,N○,20 bzw.
OPEN "R",#1,LW○+N○,20

2. Wertzuweisung

Für jede eröffnete Datei muß wenigstens eine FIELD-Anweisung der Form

FIELD #1,15 AS A○, . . .

vorhanden sein. FIELD definiert eine Maske, die der Zuweisung einzelner Bereiche des E/A-Puffers auf Variablen dient. Damit werden keine Speicherbereiche belegt oder Datentransporte vollzogen! Zu jeder Datei können beliebig viele FIELD-Anweisungen vorhanden sein. Die definierten Puffervariablen dürfen nie links vom Zuweisungsoperator "=" stehen, sie werden dann automatisch als Puffervariable gelöscht. Es ist aber möglich, Puffervariablen in PRINT- u. ae.-Anweisungen zu benutzen oder sie auf andere Zeichenkettenvariablen zuzuweisen:

PRINT A○
KETTE○=A○

Die Summe der Längen der Puffervariablen darf kleiner, aber nicht größer sein als die in der OPEN-Anweisung vereinbarte Gesamtlänge des E/A-Puffers.

Das Füllen des E/A-Puffers geschieht mit den Befehlen

LSET
RSET

LSET bedeutet linksbündiges, RSET rechtsbündiges Einschreiben von Zeichenketten. Freie Bereiche werden mit Leerzeichen gefüllt, zu lange Ketten abgeschnitten.

Beispiel:

OPEN "R",#1,"TEST.FIL",20
FIELD #1,15 AS A○,5 AS B○
LSET A○="Anzahl"
RSET B○=STR (5)

füllt den Puffer auf folgende Weise (die Schrägstriche dienen nur zur Kennzeichnung der Begrenzung):

/Anzahl   5/
M○ = "Familienname"
N○ = "Ort"
LSET A○ = M○
LSET B○ = N○
/Familienname   Ort   /

Beispiel mit zwei FIELD-Anweisungen:

FIELD #1,15 AS A○,5 AS B○
FIELD #1,20 AS C○

Nach der Zuweisung

LSET A○ = "Familienname"
LSET B○ = "Ort"

wird C○ zu

"Familienname   Ort   "

Nach der Zuweisung

LSET C○ = "Dampfschiffahrtsgesellschaft"

enthalten die Puffervariablen folgende Zeichenketten:

A○: "Dampfschiffahrt"
B○: "sgese"
C○: "Dampfschiffahrtsgese"

Der Befehl

RSET B○ = STR○(12)

erzeugt danach im Puffer den Inhalt:

"Dampfschiffahrt 12"

Besonders für lange Datensätze mit vielen Feldern ist die Verwendung mehrerer FIELD-Anweisungen einer aufwendigen Zeichenkettenverarbeitung vorzuziehen.

3. Lesen und Schreiben von Datensätzen

Die Ausgabe des Puffers auf Diskette geschieht mit der Anweisung

PUT #n,r

wobei n die Zuweisungszahl der Datei und r (r > 0) die Nummer des Datensatzes angibt.

Beispiel:

PUT #1,1

Auf die Möglichkeit, r nicht anzugeben, sondern sequentiell zu lesen oder zu schreiben, sollte zugunsten einer besseren Verständlichkeit des Programms verzichtet werden.

Das Lesen erfolgt ebenso mit der Anweisung

GET #n,r

Der Inhalt des Datensatzes steht danach in allen durch FIELD-Anweisungen definierten Puffervariablen zur Verfügung.

Durch den schnellen Zugriff auf ein definiertes Element bietet sich der Direktzugriff beispielsweise für die Verarbeitung von Arrays (Matrizen) an. Ein dreidimensionales Array V(20,5,3) könnte in BASIC folgendermaßen deklariert werden:

L=20 : M=5 : N=3
DIM V(L,M,N)
OPEN "R",#1, "ARRAY.FIL",8
FIELD #1,8 AS ZAHL○

Die Ausgabe des Arrays auf Diskette wird mit der Befehlsfolge

```
FOR I=1 TO L
  FOR J=1 TO M
    FOR K=1 TO N
      RSET ZAHL○ = STR○(V(I,J,K))
      PUT #1,((I-1)*(M*N))+(J-1)*M+N
    NEXT K
  NEXT J
NEXT I
```

realisiert.

Ein einzelnes Element des Arrays V(A,B,C) mit

1 <= A <= L,
1 <= B <= M,
1 <= C <= N

wird erreicht mit dem Befehl

GET #1,((A-1)*(M*N))+((B-1)*N)+C
V(A,B,C)=VAL(ZAHL○)

4. Schließen von Dateien

Mit dem Befehl

CLOSE bzw.
CLOSE #n

werden eröffnete Dateien geschlossen und die Bibliothekseintragungen im Directory der Diskette aktualisiert. Sollen nur einzelne Dateien geschlossen werden, wird die zugehörige Dateinummer angegeben, ohne Dateinummer werden alle zur Laufzeit eröffneten BASIC-Dateien geschlossen.

Logikbefehl



Zeichnung: Frank Steger



Börse

Inhaltsdatenbank der MP

Aufgrund der großen Nachfrage zur Inhaltsdatenbank der Zeitschrift rfe, die in rfe 11/87 zur Nachnutzung angeboten wurde, haben wir den Inhalt der nun über ein Jahr erscheinenden Zeitschrift Mikroprozessortechnik ebenfalls rechnergestützt verwaltet (Basis REDABAS).
In der Datenbank sind in Form einer Inhaltstabelle alle bisher veröffentlichten Beiträge gespeichert. Das Programm gestattet das weitere Erfassen von Beiträgen, das Suchen ausgewählter Beiträge nach Stichworten und wahlweises Ausgeben auf Bildschirm oder Drucker mit Angabe der MP-Nr., Seite und Rubrik.
Voraussetzung für den Einsatz ist ein Personal- oder Bürocomputer mit dem Betriebssystem SCP. Zur Nachnutzung wird das komplette Programmpaket einschließlich der vollständig beschriebenen MP-Inhaltsverzeichnisdatenbankdatei sowie eine Dokumentation zur Benutzung des Programms angeboten.

VEB Plasttechnik Greiz, Abt. Fertigungsmittelwirtschaft und Rationalisierungsmittel, Plauensche Str. 40–42, Greiz, 6600; Tel. 7 97 19

Heidrich

Interface-Module für Mikrorechner

Standard-Interface SI 1.2

Für den Kleincomputer KC 85/1 wurde eine Standardschnittstelle nach der Normung SI 1.2 geschaffen, die jeweils 8 Eingangs- und 8 Ausgangsdatenleitungen mit den entsprechenden Befehls- und Meldesignalen realisiert. Die Leiterplatte ist in einen Modulsteckplatz steckbar, kann aber auch über ein entsprechendes Kabel an die Kleincomputer KC 85/2 bzw. /3 angeschlossen werden. Mit diesem Interface ist z. B. die Rechnerkopplung mit dem Specord M40/M80 möglich.

Standardinterface IMS2

Es wurde eine Standardschnittstelle nach der Normung IMS2 (IEC-Bus) geschaffen, die alle Hauptfunktionen einschließlich Controller realisiert. Die Leiterplatte wurde für den Einbau in den PC 1715 konzipiert, kann aber auch mit entsprechendem Adapterkabel an alle K-1520-buskompatiblen Geräte angeschlossen werden.

V.24-Interface für elektronische Schreibmaschinen

Für die elektronischen Schreibmaschinen S6001, S6010, S6011 und die Serie S61xx wird eine Leiterplatte angeboten, die in den jeweiligen Busanschluß der Schreibmaschine gesteckt wird. Das V.24-Interface realisiert die Ausgabe auf und die Eingabe von der Schreibmaschine. Der Start des Programms des Interfaces kann von einem Taster auf der Leiterplatte oder vom angeschlossenen Computer erfolgen.

Universeller Interface-Converter UIC

Der UIC stellt ein Koppelelement zwischen verschiedenen standardisierten Interfaces der Rechentechnik dar. Der Datentransport erfolgt unidirektional von einer seriellen Schnittstelle IFSS oder V.24 auf ein paralleles Interface Centronics, IFSP, Seriendrucker 1154 oder elektronische Schreibmaschine S6005. In umgekehrter Richtung ist ein Datentransport von Centronics auf IFSS oder V.24 möglich.
Alle Betriebsbedingungen (Datenrichtung, Datenrate, Datenformat, Parität, Übertragungsprotokoll und Stromeinspeisung) sind über DIL-Schalter bzw. Wickelbrücken einstellbar.
Eine Variante mit eigenem Netzteil ist verfügbar. Bestückte Interface-Converter sind lieferbar.

8-Kanal-10-Bit-ADU

Auf der Steckeinheit im K-1520-Format befinden sich je acht Vorverstärker/Filter, S/H-Schaltungen (KP1100CK2) und A/D-Wandler (C571). Die einzelnen Kanäle werden sequentiell im Abstand von 4 µs gestartet, mit den Ausgangsdaten der A/D-Wandler erfolgt eine Parallel-Serien-Umsetzung, wobei die Daten mit 3 Adreßbits, welche den A/D-Wandlerkanal bezeichnen, und 3 Leerbits zu einem 16-Bit-Wort ergänzt werden. Die Abtastfrequenz eines Kanals beträgt 1/32 µs (30 kHz). Durch Parallelschalten von 2, 4 oder 8 Kanälen auf ein Meßsignal läßt sich die Abtastrate verzwei-, -vier- oder -achtfachen, im Maximalfall kommt man auf eine Abtastrate von 1/4 µs (250 kHz). Die Leiterplatte ist so gestaltet, daß durch Bestückungsvarianten Vorverstärker und Filter der Meßaufgabe weitestgehend angepaßt werden können und sich somit der externe Aufwand verringert. Unipolar- bzw. Bipolarbetrieb des ADU wird durch Lötbrücken auf der Leiterplatte eingestellt. Die Ausgabe der Signale erfolgt über Optokoppler potentialfrei, wobei zusätzlich noch ein Synchronsignal gebildet und gesendet wird.

Zehnertastatur für KC 85/2

Diese Tastatur ist über Klinkenstekker an KC 85/2 anschließbar, enthält die Zifferntasten 0 ... 9, Dezimalpunkt, 4 Cursortasten und die Enter-Taste.

PC-1715-Busverstärker

Dieser Busverstärker ist im Innern des PC 1715 steckbar. Der verstärkte Systembus wird herausgeführt, und somit sind K-1520-OEM-Baugruppen vom PC steuerbar.
(Einsatzgebiete: Prozeßautomatisierung, Steuerungs- und Regelungstechnik, Medizintechnik)

Friedrich-Schiller-Universität Jena, Sektion Technologie für den WGB, Technikum LAURA, Ernst-Thälmann-Ring 32, Jena, 6900; Tel. 8 22 21 31/2 21 32

Voß

Datenanalysator

Der Datenanalysator kann ebenso wie eine Station des IFLS-Netzes (entsprechend Normativmaterial der MRK für RT: Linieninterface mit serieller Informationsübertragung, MRK 82-85) z. B. in eine Prozeßrechenanlage A 6492 oder audatec eingebunden und so für folgende Zwecke genutzt werden:
– routinemäßige Überwachung der IFLS-Protokolle
– Nutzung zur Fehlersuche in Hard- und Software
– Testmittel für IFLS-Treiberentwicklungen
– Hilfsmittel bei der Programmentwicklung auf Nutzerebene.
Zum Lieferumfang gehört ein kompaktes, transportables Gerät der Abmessungen B × H × T von 310 × 80 × 270 mit einer fest installierten Software. Das Gerät enthält vier Leiterplatten: CPU-Modul (mit IFLS-Nahinterface), IFLS-Übertragungseinheit, Tastaturmodul und Netzmodul.
Die Kommunikation mit dem Bediener erfolgt über eine eingebaute Tastatur und ein anschließbares Bildschirmgerät. Als Bildschirmgerät ist ein Fernsehgerät „Junost" mit einem nachgerüsteten Anschluß für das BAS-Signal vorgesehen. Dieser Umbau kann auf Wunsch im ZfK Rossendorf durchgeführt werden. Alternativ dazu ist die Ausgabe auf einem Terminal, z. B. K 8911, verwendbar.
Die Software besteht aus einem Monitor, mit dem Programme gestartet sowie Speicherplätze angezeigt und geändert werden können. Eine HILFE-Funktion schreibt alle implementierten Programme aus. Das Programm ANALYSATOR ermöglicht dem Nutzer, einen Überblick über den IFLS-Nachrichtenverkehr zu erhalten. Mit Hilfe der Programme MASTER und SLAVE kann der IFLS-Nachrichtenverkehr aktiv beeinflußt werden.
Es sei darauf hingewiesen, daß das ZfK nur in Ausnahmefällen eine Lieferung an Industriepartner vornehmen kann.

Akademie der Wissenschaften der DDR, Zentralinstitut für Kernforschung Rossendorf, Abt. WÖK (Kolln. Seiler, kommerzielle Beratung, Tel. 5 91 20 72) oder Abt. RPR (Koll. Schmidt, fachliche Beratung, Tel. 5 91 24 61), PSF 19, Dresden, 8051

J. Schmidt

Gepackte Zahlen unter REDABAS

Bei der Verwendung des Datenbanksystems REDABAS werden in den Dateien alle numerischen Werte in der üblichen ASCII-Darstellung, d. h. mit zwei Byte, abgespeichert. Zur eindeutigen Darstellung der möglichen Zeichen von Zahlen (Ziffern: 1 . . . 9, Sonderzeichen: + – .) würde ein Halbbyte ausreichen. In einigen Softwaresystemen wird diese Tatsache genutzt, indem bei numerischen Werten je zwei Ziffern bzw. Sonderzeichen in einem Byte zusammengepackt werden. Leider ist bei REDABAS diese Funktion zum Packen und Entpacken von numerischen Feldern nicht realisiert. Dieses Verfahren mit den in REDABAS implementierten Funktionen zu realisieren, ist aus Verarbeitungszeitgründen nicht vertretbar. Von uns wurde ein Verfahren entwickelt, das diese Funktion auf jeder in der DDR verfügbaren 8-Bit-Rechentechnik unter dem Betriebssystem SCP/CPA/CPM mit dem Datenbanksystem REDABAS ermöglicht. Es können sowohl Datenbankfelder als auch zugehörige Indexdateien mit numerischem Indexbegriff in diesem Format verarbeitet werden.
Grundlage dieses Verfahrens ist die Einbindung eines Assembler-Programmes in den Programmablauf des Hochsprachenteils. Die Laufzeit dieses Moduls ist unmerklich und liegt im ms-Bereich.
Die Anwendung dieses Moduls bringt dem Nutzer bei reiner Zahlenspeicherung eine 50prozentige Einsparung seines Diskettenbedarfs. Bei größeren Projekten sind Diskettenwechsel nur noch halb so oft nötig. Bei Verwendung von Datenbanken, die unter dem Betriebssystem SIOS erstellt worden sind, ist eine einfache Wandlung bzw. Verwendbarkeit der gepackten Daten möglich.
Die Assemblerroutine wird über eine POKE-Zeile an eine beliebige freie Stelle im Hauptspeicher gebracht. Von dort kann sie über den CALL-Befehl aufgerufen werden.
Zum Lieferumfang gehört außer dem Assembler-Listing auch ein Demonstrationsprogramm in REDABAS, das die Anwendungsmöglichkeit demonstriert.
Der Preis für die Nachnutzung einschließlich Dokumentation beträgt ca. 140 Mark.

VEB Kreisbetrieb für Landtechnik Burg, BfN, Königsborn, 3101

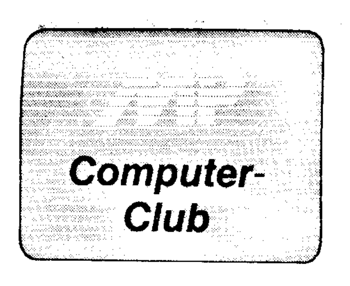
Northe

Schnittstellenkonverter

In der z. Z. realisierten Variante ermöglicht der Konverter die Konvertierung von V24 nach SIF 1000 und von V24 nach IFSS.
Dabei besteht die Möglichkeit der automatischen Bitrateneinstellung sowie die Auswahl verschiedener Protokollvarianten und Signalpegel (SIF 1000) über DIL-Schalter. Zusätzlich ist die Messung und Anzeige der Bitraten der beiden seriellen Schnittstellen möglich.
Durch Hinzufügen weiterer Softwaremodule ist leicht eine Funktionserweiterung möglich.
Das Gerät kann auch für andere Anwendungsfälle eingesetzt werden. So ist z. B. der Einsatz als Steuermodul, mit der Möglichkeit der Ankopplung an einen Leitrechner über eine serielle Schnittstelle, denkbar.
In der jetzigen Anwendung soll das Gerät hauptsächlich den Anschluß verschiedener Druckertypen (SD 1154, SIF 1000; SD 1152, IFSS) an den PC 1715 ermöglichen.

Technische Universität Karl-Marx-Stadt, Sektion Automatisierungstechnik, PF 964, Karl-Marx-Stadt, 9010

J. Mehnert



Selbststart von BASIC-Programmen

Diese Anregung ermöglicht den Selbststart von BASIC-Programmen sofort nach dem Einlesen vom Tonband, ohne vorher den BASIC-Interpreter aufrufen, die Frage "MEMORY END : " beantworten und das Programm über CLOAD "Name" einlesen zu müssen.

Diese Methode könnte sinnvoll angewendet werden bei Steuerungen, die ohne Monitor auskommen, der aber zur Fehlererkennung bei der Eingabe nötig ist. Der Monitor kann nun entfallen, weil nur noch wenige, kurze Eingaben notwendig sind:

– Im >OS-Modus Programmname (ohne "") eingeben
– ENTER-Taste betätigen (auf Bildschirm: start tape)
– Tonband starten
– ENTER-Taste betätigen

Nach dem Einlesen startet sich das Programm selbst, ohne eine weitere Taste betätigen zu müssen.

Diese hier vorgestellte Anregung ist ohne weitere Bedingungen auf dem KC 87 lauffähig. Bei KC 85/1 bzw. Z 9001 ist der BASIC-Interpreter (ROM) zu stecken.

Das vorhandene BASIC-Programm ist vorher wie ein Maschinenprogramm auszulagern. Das Programm muß außerdem den kompletten BASIC-Arbeitsspeicher enthalten und den zusätzlichen Befehl: Lade Tastaturpuffer mit dem Code für die RUN-Taste.

Um ein BASIC-Programm als MC-File auszulagern, muß die Anfangs-, die End- und die Startadresse bekannt sein.

Im BASIC-Arbeitsspeicher steht auf den Adressen 983/984 die Anfangsadresse der nächsten BASIC-Zeile bei der Neueingabe eines Programmes, was bei einem fertigen Programm die erste freie Speicherzelle hinter dem Programm ist. Somit kann die gesuchte Endadresse durch: PRINT DEEK (983) –1 und eine anschließende dezimal-hexadezimal Umrechnung ermittelt werden.

Ein BASIC-Programm beginnt bei Verwendung eines ROM-Interpreters auf der Adresse 401H. Ab 300H bis 400H steht der BASIC-Arbeitsspeicher. Auf den dem Arbeitsspeicher vorangehenden Adressen kann man das Laden des Tastaturpuffers eintragen. Dort steht dann:

3E 1D LD A,1DH ;Lade Akku mit Code für RUN
32 25 00 LD (25H),A ;Lade Tastaturpuffer mit Akku

Das kann mit dem Zusatzmonitor (ZM) realisiert werden. Mit dem S-Befehl wird der Maschinencode der obigen Befehle (3E, 1D, 32, 25, 00) ab der Adresse 2FBH eingetragen. Im BASIC geschieht das durch das Poken der Werte 62, 29, 50, 37 und 0 der Reihe nach ab der Adresse 763 bis 767.

Die Anfangs- und die Startadresse ist also 2FBH. Jetzt kann das Programm in gewohnter Weise als MC-Programm mit der Anfangs- und Startadresse 2FBH und der nach oben beschriebener Methode gefundener Endadresse ausgelagert werden.

Bei dieser Methode wird die Struktur des BASIC-Programmes nicht verändert und kann nach dem Einlesen wie jedes andere Programm behandelt werden. Die einmal geschriebenen Befehle ab der Adresse 2FBH stehen bis zum Ausschalten des Computers bereit. Auch nach dem Einlesen eines wie oben behandelten Programmes sind diese Speicherplätze wieder geladen und müssen vor einem erneuten Auslagern nicht wieder eingetragen werden.

Zusammenfassend sind also folgende Arbeitsgänge notwendig:

– BASIC-Programm laden
– Endadresse feststellen: PRINT DEEK (983) –1; Umrechnung in Hexadezimalzahl
– Laden des RUN-Befehles eintragen: Zellen 763 bis 767 mit Bytefolge 62, 29, 50, 37, 0 über POKE-Befehl laden
– BASIC-Interpreter verlassen: BYE
– Mit Zusatzmonitor (ZM) oder OS-SAVE das erweiterte BASIC-Programm als MC-Programm auf Band speichern:

SAVE name 2 FB Endadresse (hexadezimal) 2 FB

Diese kleine Mühe lohnt sich, denn bei ständigen Arbeiten mit BASIC-Programmen vereinfacht sich das Laden der Programme wesentlich.

Wolfgang Meixner

MC-Programme in BASIC-Programme eingebettet

Häufig ist es notwendig, aus BASIC-Programmen heraus MC-Programme aufzurufen, um eine spürbare Erhöhung der Rechengeschwindigkeit zu erreichen. Diese müssen dafür gleichzeitig mit im Speicher untergebracht sein. Das Trennen des verfügbaren RAM-Bereiches in BASIC-Arbeitsspeicher und Bereich für die MC-Programme (hinter dem Zeichenketten-Speicherbereich), das getrennte Laden der beiden Programmarten oder auch das Einlagern der MC-Programme in den Freibereich des BASIC-Arbeitsspeichers sind umständlich und fehleranfällig. Das betrifft sowohl die Vorbereitungs- als auch die Nutzungsphase.

Wünschenswert wäre, BASIC- und die dazugehörigen MC-Programme in einem einmaligen Arbeitsgang zu verbinden, so daß sie mit einem Ladebefehl geladen werden können und sich während der Laufzeit nicht mehr ungewollt beeinflussen. Das kann bei Robotron-Kleincomputern erreicht werden durch die Abspeicherung der MC-Programme vor dem Bereich für BASIC-Programme, d. h. zwischen Notizspeicher und Quellprogrammbereich des Interpreters (Bild 1).

Die Voraussetzungen dafür sind:

– Der BASIC-Interpreter „orientiert" sich bei der Verwaltung seines Arbeitsspeichers am Zeiger PSTBEG, der beim Neustart auf 401H initialisiert wird und nach hinten verschieblich ist. Beim Neustart werden außerdem SVARPT, DVARPT und FSLPTR auf (PSTBEG)+2 gestellt, und der Speicher wird initialisiert mit 3× Null ab (PSTBEG)–1.

– Das Kommando CSAVE „name" lagert immer den Bereich zwischen 41H (Initialisierungswert für PSTBEG) und (SVARPT) als BASIC-Programm auf Kassette aus. CLOAD „name" lädt ein ausgelagertes BASIC-Programm ab (SVARPT)–2 und setzt die Zeiger SVARPT, DVARPT und FSLPTR hinter den geladenen Programmteil.

– Auf Adresse (PSTBEG) beginnt das BASIC-Programm, der davorliegende Speicherplatz muß den Wert 0 besitzen, damit eine korrekte Programmabarbeitung möglich ist.

Die einzelnen Arbeitsschritte, um MC-Programme nach diesem Prinzip einzubetten, sind:

1. Erstellen der MC-Programme mit einer Anfangsadresse von minimal 414H (1044 dez.) und Bestimmen der Länge. Die MC-Programme können auf Kassette unter name1 ausgelagert und später zum Verbinden mit dem BASIC-Programm geladen werden (Pkt. 7).

2. Neustart des BASIC-Interpreters

3. Bestimmen der Anfangsadresse des eigentlichen BASIC-Programms name2 (z. B. 2000) und Initialisierung des Speichers mit FORI = 0TO2 : POKE 2000 + I, 0 : NEXT

4. Eintragen der Zeigerumstellung für PSTBEG mit

10 DOKE 863, 2001 : RUN

5. Zeigerumstellung für die Erarbeitung des eigentlichen BASIC-Programms name2 (SVARPT, DVARPT, FSLPTR, PSTBEG) mit

DOKE 983, 2003:DOKE 985, 2003
DOKE 987,2003:DOKE 863,2001

6. Erarbeiten des eigentlichen BASIC-Programms oder Einlesen mit CLOAD „name2"

7. Einlesen der MC-Programme von Kassette (Pkt. 1) mit

BYE
CLOAD name1
WBASIC

8. Gemeinsames Auslagern von BASIC- und MC-Programmen auf Kassette (Bereich von 401H... (SVARPT)) mit

CSAVE „name3"

Wenn nach einem Neustart des Interpreters das Programm mit CLOAD „name3" geladen und mit RUN gestartet wird, wird Zeile 10 aus Pkt. 4 abgearbeitet, die den Zeiger PSTBEG umstellt und das eigentliche BASIC-Programm startet.

Zu beachten ist, daß unmittelbar nach dem Laden von name3 mit LIST nur die Zeile 10 (Pkt. 4) zu sehen ist sowie nach RUN nur noch das eigentliche BASIC-Programm (Pkt. 6) und die Zeile 10 nicht mehr, da LIST nur den Bereich zwischen (PSTBEG) und Programmende (000) erfaßt.

Dr. Hans-Jürgen Busch

Speicherbereich	Zeiger
Notizspeicher (Systemzellen BASIC)	← fest, 300H
Quellprogramm	← PSTBEG (35FH)
Tabelle der einfachen Variablen	← SVARPT (3D7H)
Tabelle der Feldvariablen	← DVARPT (3D9H)
frei	← FSLPTR (3DBH) } FRE (A)
Stack	← SP (Stackpointer)
ZK-Speicherbereich	← STDPTR (356H) } FRE(A$)
	← SWAPTR (3C4H)
	← MEMSIZ (3B0H)

Bild 1 KC-Speicheraufteilung

Spracheingabemodul zum KC 87

Zur Erweiterung der Einsatzmöglichkeiten des KC 87 wurde im VEB Robotron-Meßelektronik „Otto Schön" Dresden ein Modul zur Spracheingabe entwickelt. Die menschliche Sprache als Eingabemedium für Computerbedienung ist ein anzustrebender Idealzustand. Neben einer hervorragenden Rationalisierung der Arbeit mit dem Computer würden sich völlig neue Einsatzgebiete erschließen. Für perfekte und sprachunabhängige Spracherkennung ist jedoch eine aufwendige Elektronik und umfangreiche Software erforderlich, die den Rahmen eines Kleincomputers hinsichtlich Kosten und Größe bei weitem übersteigt.

Bei der Entwicklung des Spracheingabemoduls (SE-Modul) zum KC 87 wurde aus wissenschaftlichen Forschungsergebnissen der Technischen Universität Dresden eine hardwaremäßig einfache Version abgeleitet. Durch eine leistungsfähige Software werden dennoch sehr gute Leistungsparameter erreicht:

Wortschatz
max. 50 Wortklassen
Anzahl der Referenzmuster:
max. 200 im Arbeitsspeicher des Computers
Wortlänge:
0,2 ... 1,8 s
Pause zwischen zwei Worten:
minimal 0,2 s.

Der Wortschatz wird in einem Arbeitsgang „Lernen" in den Computer eingegeben und dort gespeichert. Dieser Wortschatz ist sprecherspezifisch. Das Anlernen des Wortschatzes erfordert ein gewisses Training und disziplinierte, reproduzierbare Aussprache. Die Erkennungsrate hängt wesentlich vom kooperativen Verhalten des Sprechers, dem phonetischen Abstand der zu erkennenden Worte, den Umgebungsgeräuschen und der exakten Durchführung des Lernvorganges ab. Es werden Erkennungsraten von 90 Prozent erreicht. Das Erkennungsergebnis wird als Ordnungsnummer der erkannten Wortklasse in einer Arbeitszelle des Computers zur Verfügung gestellt.

Hardware des Spracheingabe-Moduls

Der SE-Modul wird im Modulschacht des Kleincomputers betrieben. Er enthält die Baugruppen einschließlich des Mikrofonanschlusses. Das Sprachsignal wird mittels eines mitgelieferten piezokeramischen Mikrofons (Sprechzeug SP 75 vom VEB Funkwerk Kölleda) in ein analoges elektrisches Signal gewandelt. Nach dem Durchlaufen einer Verstärkerstufe wird das Signal parallel einem Hochpaß und einem Tiefpaß zugeleitet. Die Schnittfrequenz beträgt 1 kHz. Die beiden Ausgangssignale der Filterstufen werden auf TTL-Pegel normiert und einem Zeitgeber-Baustein (CTC) zugeführt, der die

Abstände zwischen den Nulldurchgängen der Analogsignale bestimmt und für die weitere Softwareverarbeitung zur Verfügung stellt.

Software

Die umfangreiche Softwarebearbeitung umfaßt ein Programm von etwa 4 KByte. Sie enthält die komplexe Signalanalyse, Referenzmusterspeicherung und Worterkennung sowie eine umfangreiche Bedienerführung in Menütechnik.

Signalanalyse und Worterkennung

Aus den von der CTC bereitgestellten Meßwerten für die Zeitintervalle zwischen den Nulldurchgängen werden in äquidistanten Zeitfenstern von 25 ms Nulldurchgangshistogramme für 8 Frequenzklassen (je 4 pro Filterbereich) gebildet. Das Ergebnis entspricht in grober Näherung einem 8-Kanal-Kurzzeit-Filterspektrum.
Anschließend erfolgt eine Längennormierung auf 16 Zeitfenster, d. h. 16 Byte. Diese 16 Byte stellen dann das Wortmuster dar, das unter einer Klassen-Nummer im Arbeitsspeicher des Computers abgelegt werden kann. In einem Vergleichsprozeß zu den bereits im Speicher vorhandenen Referenznummern wird der Grad der Ähnlichkeit ermittelt und danach
- ein Erkennungsergebnis ausgegeben, wenn ein sehr ähnliches Muster bereits existiert
- oder zu einer Wortklasse ein weiteres Muster angelegt, wenn eine mittlere Ähnlichkeit ermittelt wird
- oder ein Muster einer neuen Wortklasse angelegt, wenn keine oder nur geringe Ähnlichkeit existieren.

Im Spracherkennungsprozeß wird dann immer die Klassennummer des ähnlichsten Referenzmusters als Erkennungsergebnis ausgegeben. Überschreitet das Abstandsmaß zu allen Referenzmustern einen maximalen Wert, wird eine Zuweisung abgelehnt und das Wort zurückgewiesen. Diese Rückweisung wird durch ein akustisches Signal angezeigt.

Bedienerführung

Dem Nutzer wird jederzeit durch eine komfortable Bedienerführung die Möglichkeit des Eingriffs in den Programmablauf gegeben.
In einem Anfangsmenü kann zwischen unterschiedlichen Teilprogrammen für
- Anlernen eines neuen Wortschatzes
- Laden eines bereits gelernten Wortschatzes vom Magnetband
- Abspeichern eines angelernten Wortschatzes auf Magnetband
- Löschen und Umbenennen eines Wortschatzes

ausgewählt werden. Dem Nutzer wird stets die Anzahl der im Speicher bereits abgelegten Referenzmuster angezeigt. Neue Wortmuster werden nach Zuweisung einer Wortklassen-Nummer und Quittierung über die Tastatur im Speicher abgelegt.

Praktische Anwendung

Neben allgemeinen Anwendungsfällen des Spracheingabemoduls für experimentelle Arbeiten und Computerspiele hat sich dieser Modul in der Erprobungsphase bei zwei sehr wichtigen Einsatzaufgaben bewährt:
1. Einsatz als „phonetische Tastatur" zur Computerbedienung an verschmutzten Arbeitsplätzen ohne Tastaturberührung. Die erforderlichen Computerkommandos, z. B. die Bewegungskommandos für einen Manipulator, werden durch menschliche Sprache eingegeben. Um Fehlfunktionen des Computers, hervorgerufen durch falsch erkannte Kommandos, zu vermeiden, wird das erkannte Kommando erst auf dem Bildschirm angezeigt und nur nach einem weiteren „Gültig"-Kommando ausgeführt.
2. Einsatz des Spracheingabemoduls bei der Spracherziehung gehörloser oder schwerhöriger Schüler. Gehörlose Schüler benötigen für Sprachübungen stets einen Lehrer, der die Sprachleistung begutachtet, da der Schüler aufgrund seines Gehörverlustes selbst nicht dazu in der Lage ist. Der Computer kann wesentliche Teile der Funktion des Lehrers übernehmen. Unter Aufsicht lernt der Schüler den zu übenden Wortschatz dem Computer an und kann dann allein mit dem Computer üben. Nur sauber gesprochene Worte erkennt der Computer an und registriert so die Übungsergebnisse des Schülers. Neben der besonderen Motivation der Schüler bei der Lernarbeit am Computer tritt somit eine Entlastung des Lehrers von Routinearbeiten und gleichzeitig eine Vervielfachung des vorhandenen Übungszeitfonds ein.
Der Spracheingabemodul 690 032.4 ist ohne jegliche Veränderung auch im KC 85/1 arbeitsfähig. Zum Betrieb ist ein RAM-Erweiterungsmodul erforderlich, in den das Steuerprogramm von der mitgelieferten Magnetband-Kassette R0113 eingelesen wird.

Dr. Gunter Kleinmichel
VEB Robotron-Meßelektronik
„Otto Schön" Dresden

Zeichenketteneingabe beim KC 85/3

Wird bei der Eingabe von Zeichenketten (Strings) ein Komma eingegeben, so meldet der Computer 'EXTRA IGNORED', und die Zeichen nach dem Komma gehen verloren. Manche BASIC-Interpreter (z. B. beim SCP) besitzen dafür die LINE INPUT-Anweisung zur Eingabe bis zur ENTER-Taste. Eine Möglichkeit, dies zu simulieren ist, als erstes Zeichen ein Anführungszeichen (") einzugeben. Dieses Verfahren ist ja von DATA-Zeilen bekannt, wenn dort Zeichenketten, Doppelpunkt oder Komma enthalten sein sollen. Das kann aber leicht beim Eingeben vergessen werden. Ein Ausweg ist hier, die INPUT-Anweisung zu modifizieren. Das folgende Programm dient dazu:

```
10 DATA 229,33,34,32,34,98,3,33,0,
32,34
20 DATA 100,3,33,99,3,195,178,197
30 FOR A=0TO18:READ X:POKE
512+A,X:NEXT
40 AD=843:AN=512:AA=-14930
```

Die DATA-Zeile enthält ein Maschinenprogramm, welches die INPUT-Anweisung so abändert, daß das Anführungszeichen auch ohne Eingabe im Puffer steht. Dieses Programm wird im Beispiel auf die Adresse 200H „gepokt". Da der Aufruf der INPUT-Anweisung über einen Sprung im RAM (34AH) erfolgt, kann vom Programm aus auf den LINE INPUT-Modus umgeschaltet werden.
Die Variable AD enthält die Adresse, auf die der Sprung zur Eingabe eingetragen wird. Die Variable AN enthält die LINE INPUT-Adresse, AN ist die normale Eingabe.
Soll die Zeileneingabe aufgerufen werden, muß erst umgeschaltet werden. Nach der Eingabe sollte sofort zurückgeschaltet werden, da sonst auch keine Zahleneingabe möglich ist.
Dies kann geschehen:

```
100 DOKEAD,AN
110 INPUT "KENNZEICHEN : ";K$
120 DOKEAD,AA
130 INPUT"ANZAHL :";Z
```

Für die Abspeicherung der Zeichenketten auf Kassette stehen zwei Möglichkeiten zur Verfügung:
- **Datenfelder mit CSAVE * /CLOAD ***
- ASCII-Dateien mit PRINT#**1/INPUT#1**

Die letztere hat den Vorteil, daß beim Laden sofort die Daten herausgesucht werden können, die benötigt werden. Somit müssen umfangreiche Dateien nicht immer komplett in den Speicher geladen werden.
Beim Laden von ASCII-Dateien kann es aber wieder zum oben beschriebenen Fehlerbild kommen. Hier kann jedoch schon bei der Aufzeichnung das Anführungszeichen angefügt werden. Die Programmzeilen von 200 bis 300 verdeutlichen das:

```
200 OPEN 0#1 "PROG1"
210 PRINT#1 CHR$(34);"NAME,VOR-
NAME"
220 PRINT#1 CHR$(34);X$

300 CLOSE 0#1
```

Geladen wird dann folgendermaßen:

```
500 OPEN I#1 "PROG1"
510 INPUT#1 T$

600 CLOSE 0#1
```

Klaus-Dieter Kirves

Schnelles Bildschirmlöschprogramm für KC 85/3

Durch die komplizierte Organisation des Bildwiederholspeichers im KC 85/3 bzw. KC 85/2 dauert die Bildschirmlöschung über Shift-Home und CLS in BASIC relativ lange. Eine schnellere Variante bietet das folgende kleine Maschinenprogramm.

Arbeitsweise

Nach Aufruf des Programms werden folgende Teilfunktionen realisiert:
1. Beschreiben des Pixelspeichers (8000H bis A7FFH) mit 0H
2. Beschreiben des Videospeichers (B200H bis B6FFH) mit 0H
3. Beschreiben des Farbattributspeichers mit der aktuellen Farbinformation aus der Arbeitszelle COLOR (B7A3H)

Es ist auch möglich, die Teilfunktion 3 allein aufzurufen. Hierbei kann ohne Beeinflussung des Bildschirminhaltes die Bildschirmfarbe verändert werden.

Besonderheiten

Das Programm löscht stets den gesamten Bildschirm. Der Cursor steht nach der Löschung in der linken oberen Ecke des aktuellen Fensters.

Programmaufruf

In BASIC wird mit dem CALL-Befehl aufgerufen, im angegebenen Ladebereich der Routine (siehe Bild 1) z. B.:
- Bildschirmlöschung mit CALL 0 D (CALL* 0H)
- Farbveränderung mit CALL 57 D (CALL* 39H).

Das Programm ist jedoch auch auf allen anderen RAM-Bereichen lauffähig, es sind dann lediglich die Einsprungadressen neu zu ermitteln.

```
0000  CD 18 F0 21 00 80 0E 40
0008  3E 00 B1 28 0C 3E 00 0D
0010  06 A0 36 00 23 10 FB 18
0018  EF 21 00 B2 0E 28 3E 00
0020  B1 28 0B 3E 20 0D 06 20
0028  77 23 10 FC 18 F0 21 A0
0030  B7 3E 00 77 23 77 CD 1B
0038  F0 CD 18 F0 21 00 A8 0E
0040  28 3E 00 B1 28 0C 3A A3
0048  B7 0D 06 40 77 23 10 FC
0050  18 EF CD 1B F0 C9 FF FF
```

Bild 1 Maschinenprogramm

```
 10 REM ROUTINE LADEN
 20 DATA 205,24,240,33,0,128,14,64
 30 DATA 62,0,177,40,12,62,0,13,6,160
 40 DATA 54,0,35,16,251,24,239,33,0,178
 50 DATA 14,40,62,0,177,40,11,62,32,13
 60 DATA 6,32,119,35,16,252,24,240,33
 70 DATA 160,183,62,0,119,35,119,205
 80 DATA 27,240,205,24,240,33,0,168
 90 DATA 14,40,62,0,177,40,12,58,163
100 DATA 183,13,6,64,119,35,16,252,24
110 DATA 239,205,27,240,201
120 FOR I=0 TO 85
130 READ Z
140 POKE I,Z
150 NEXT
160 REM FARBTEST
170 CALL 0
180 FOR I=0 TO 7
190 FOR K=0 TO 31
200 PRINT AT(10,10);"F A R B T E S T"
210 PRINT AT(12,10);"VORDERGRUND: ";K
220 PRINT AT(14,10);"HINTERGRUND: ";I
230 COLOR K,I
240 CALL 57
250 PAUSE 20
260 NEXT
270 NEXT
280 COLOR 7,1
290 CALL 0
300 END
```

Bild 2 BASIC-Beispiel

Programmbeispiel

Bild 2 zeigt ein Demonstrationsbeispiel für die Anwendung des Programms. In den Zeilen 10 bis 150 wird das Programm in den Speicherbereich 0000H bis 0055H „hineingepokt" (dezimal 0 bis 85).
Nach Löschung des Bildschirmes in Zeile 170 (CALL 0) werden alle möglichen Farbkombinationen zwischen Vorder- und Hintergrundfarbe ohne Veränderung des Bildschirminhaltes durchlaufen (Zeilen 230/240).
Die durch den COLOR-Befehl voreingestellte Farbe wird mit dem CALL-Befehl sofort übernommen.
Die Befehle auf den Zeilen 280/290 stellen die ursprüngliche Farbkombination (Vordergrund weiß/Hintergrund blau) wieder her und löschen den Bildschirm.
Neben der Möglichkeit, das Programm von BASIC aus mittels POKE-Befehl in den Speicher zu schreiben, kann man das Maschinenprogramm natürlich auch im Betriebssystem mit MODIFY einspeichern und mit SAVE zur universellen Nutzung retten.

Hans Langenhan, Ohrdruf

Zu FABAS

In Heft 12/1987 ist das BASIC-Programm FABAS zwar richtig, jedoch sind uns im erläuternden Text Fehler unterlaufen. Entsprechend den Programmzeilen 320...350 lauten im Abschnitt „Routinen des Maschinen- und BASIC-Programms" die Call-Adressen für

Subtraktion	**4CF**
Multiplikation	**531**
Division	**5CC**
Prozentrechnung	**5CF**

Bei einem Fehler ist (RS) nicht Null, sondern (RS)<>0.
Im Abschnit „Variablen und Hilfsprogramme" sind die Lokalen Variablen für Eingabe:
A$ enthält Zahl als ASCII-Kette: **A,B,C,D,I,L,X,Y,S$.**

Autor und Redaktion

Fachtagung Computer- und Mikroprozessortechnik '87

Am 8. und 9. Dezember 1987 fand diese Veranstaltung in Magdeburg mit über 550 Teilnehmern statt. Veranstalter der jährlich durchgeführten Tagung war die wissenschaftliche Sektion Computer- und Mikroprozessortechnik im Fachverband Elektrotechnik der Kammer der Technik in Verbindung mit dem Fachausschuß „Mikroprozessor-Interfacesysteme", dem KDT-Bezirksvorstand Magdeburg und der Sektion Automatisierungstechnik und Elektrotechnik der TU „Otto von Guericke" Magdeburg.
Die Fachtagung vermittelte in 8 Hauptvorträgen, 21 Vorträgen in vier Sektionen und 16 Posterbeiträgen umfassende Informationen über Stand und Perspektiven der Computer- und Mikroprozessortechnik sowie ihrer Anwendung und über aktuelle Probleme der Schaltkreisentwicklung unter besonderer Berücksichtigung der Anforderungen in der DDR.
Inhaltlich war sie in folgende **Schwerpunktkomplexe** gegliedert:
- Meßwerterfassung und Prozeßautomatisierung
- Mikrorechner (Hard- und Software)
- Bussysteme und ihre Anwendung
- Anwendungsspezifische Schaltkreise (ASICs).

Zu jedem dieser vier Themenkomplexe wurden zwei Hauptvorträge gehalten.
Prof. Dr. Neumann von der TU Magdeburg gab einen Überblick zum Stand und Trend der Mikrorechner-Automatisierungsanlagen für die Mittel- und Großautomatisierung. Hauptgesichtspunkte zur Beurteilung des gegenwärtigen Standes und Trends sind die rechentechnische Basis, die Datenübertragungslösung, Schnittstellen zum Prozeß und zum Menschen sowie die Gewährleistung einer hohen Verfügbarkeit. Die Weiterentwicklung wird vor allem durch die rechentechnische Basis (Übergang auf 32-Bit-Rechner, Erhöhung der Speicherkapazität, verstärkte Nutzung von Fach- und Hochsprachen, Wissensverarbeitung) sowie durch die Anwendung standardisierter Kommunikationssysteme, vor allem von Feldbussystemen, vorangetrieben.
Einen Überblick zu intelligenten Funktionseinheiten mit Feldbusanschluß für die Meßwerterfassung und -vorverarbeitung gaben *Prof. Dr. Seifart* und *Doz. Dr. Michaelis* (beide TU Magdeburg). Die bisher vorliegenden Erfahrungen zeigen, daß die meisten Meß- und Automatisierungsaufgaben mit einem kleinen Sortiment programmierbarer Prozeßkoppelmodule lösbar sind, wobei die jeweilige Anpassung an den Einsatzfall häufig ausschließlich durch die Gestaltung der Anwendersoftware erfolgt. Als realisierte Beispiele wurden ein universeller programmierbarer Prozeßkoppelmodul (PPM) mit direktem Sensoranschluß für analoge und binäre Eingangssignale, ein Innenrechner für Korbverseilmaschinen und eine Baugruppe zur Meßwerterfassung mit CCD-Bildsensoren vorgestellt.
Prof. Dr. Roth (TH Ilmenau) erläuterte in seinem Vortrag „Mikroprozeßrechner und Expertensysteme – perspektivische Entwicklungen" die große Bedeutung zukünftiger wissensverarbeitender Automaten für die weitere Entwicklung der wissenschaftlich-technischen Revolution. Sowohl für die industriellen als auch für die intellektuellen Arbeitsmittel vollzieht sich die Entwicklung vom Signalniveau (U,I,Q...) über das syntaktische (Zeichen, Zeichenketten...) und semantische (Bedeutung, Wissen...) hin zum pragmatischen (Zweck, Ziel, Sinn, Effekt...) Niveau. Aus dem gemeinschaftlich verlaufenden Prozeß der zusammenhängenden Erkenntnismittel (Expertensysteme) und Produktionsmittel (Automaten) leitete der Vortragende eine dreistufige Gliederung qualitativ unterscheidbarer Automaten ab: nichtintelligente Automaten als „Turing-Maschinen", elementarintelligente Automaten als „Leibnitz-Maschinen" und kreativintelligente Automaten als „Wiener-Maschinen" (Selbstorganisation).
Im Vortrag „Leistungsgesteigerte 16- und 32-Bit-Mikroprozessorsysteme" gab *Prof. Dr. Neubert* (TU Dresden) einen Überblick zur Mikroprozessorentwicklung und ging detaillierter auf die Prozessoren 8086, 80286, 80386 und einige weitere Systeme ein. Eingehend wurden die Unterschiede, u. a. in der inneren Struktur zwischen den Typen 8086, 80286 und 80386 erläutert.
Abschließend verwies der Vortragende auf den Trend, daß immer mehr Konzepte bei Mikroprozessoren realisiert werden, die bisher den großen EDVA vorbehalten waren und daß sich die Entwicklungslinien der Mikroprozessoren weiter annähern.
Zum Komplex „Bussysteme" referierte *Prof. Dr. Meiling* (TU Dresden) über den Einfluß der Prozessorarchitektur auf die Struktur von Rechnerbussystemen. Einen besonderen Schwerpunkt bildeten Betrachtungen zur Erhöhung des Befehlsdurchsatzes durch zweckmäßige Rechnerarchitekturen und Buskonzepte. Eingehender ging der Vortragende auf die 16- bzw. 32-Bit-Bussysteme Multibus I und II sowie den VME-Bus ein.
Prof. Dr. H. Löffler (Informatikzentrum an der TU Dresden) vermittelte in seinem Übersichtsvortrag „Lokale Netze" einen Überblick über den Aufbau, die Wirkungsweise und die Bedeutung serieller Kommunikationsverbindungen zur Rechnerkopplung.
Prof. Dr. Müller (TU Karl-Marx-Stadt) sprach zum Thema „Entwurf und Anwendung von Gate-Array-Schaltkreisen unter dem Aspekt der Zusammenarbeit von Anwender und Hersteller". Eingehend betrachtete er Varianten von zweckmäßigen Schnittstellen zwischen Anwender und Hersteller der Schaltkreise und setzte sich leidenschaftlich dafür ein, daß die Anwenderindustrie die Möglichkeiten der Gate-Array-Technik möglichst umfassend nutzen möge.
Dr. Gieseler (VEB Forschungszentrum Mikroelektronik Dresden) referierte zum Thema „Entwurf und Anwendung von Gate-Array-Schaltkreisen unter dem Aspekt der mit U 5200 erreichten technischen Parameter". Zielstellung der DDR-Halbleiterindustrie ist es, in jedem Technologieniveau auf Basis der jeweiligen Standardtechnologie ein Gate-Array-System zu entwickeln. Beim System U 5200 kommt eine 4-µm-CMOS-Technologie mit zwei Polysiliziumebenen und einer Metallisierungsebene zur Anwendung. Eingehend wurden das Entwurfssystem „Archimedes", der Master U 5201 sowie statische und dynamische Parameter des Systems beschrieben.
Die Themen und Autoren der in den vier Fachsektionen gehaltenen 21 Vorträge und 16 Posterbeiträge wurden bereits im Heft 11/87 der MP aufgeführt (S. 351). Daher wird nachfolgend lediglich ein kurzer zusammenfassender Überblick gegeben.
In der Sektion A – Meßwerterfassung und Prozeßautomatisierung – stand der Einsatz von Mikrorechnern zur Meßwerterfassung im Mittelpunkt. Neben feldbuskoppelbaren intelligenten Modulen, der Kopplung von Meßtechnik mit BC, PC und KC, optoelektronischer Meßtechnik und Computergrafik wurden Aspekte der Prüfstrategie für elektronische Erzeugnisse behandelt.
Die Sektion B – Mikrorechner (Hard- und Software) – beinhaltete Vorträge über den Stand und Trend der Mikrorechentechnik in der DDR. Einen besonders großen Teilnehmerkreis wiesen dabei die beiden Vorträge des Kombinates Robotron zum neuen ESER-PC EC 1834 auf. Weitere Schwerpunkte waren das Programmier- und Entwicklungssystem P 8000, 16-Bit-Single-board-Computer und Multirechnersysteme.
In der Sektion C – Bussysteme und ihre Anwendung – bildeten Feldbussysteme, lokale Netze, Probleme der Busvergabesteuerung, Systembusprüftechnik und Simulation von Bussystemen einen Schwerpunkt.
In der Sektion D – Kundenspezifische Schaltkreise – wurde über aktuelle Erfahrungen beim Entwurf und dem Einsatz des Gate-Array-Schaltkreissystems U 5200 im Schwermaschinenbau, zur Überwachung und Steuerung von Gasbrennern, zur Signalauswertung in Meßsystemen mit CCD-Zeilen sowie über die Simulation von Gate-Array-Schaltkreisen berichtet.
Die Kurzfassungen der Vorträge und Posterbeiträge sind in einem Tagungsband (84 S.) zusammengefaßt.
Die nächste Veranstaltung Computer- und Mikroprozessortechnik '88 findet vom 15. bis 16. Dezember 1988 in Berlin statt.

Prof. Dr. Seifart

Systems '87 Computer und Kommunikation

10. Internationale Fachmesse und Internationaler Anwenderkongreß
Auf dem Münchner Messegelände fanden über 1270 Aussteller aus 20 Ländern in 25 Hallen mit fast 107 000 m^2 Ausstellungsfläche sehr gute Bedingungen, um wissenschaftlich-technische Ergebnisse der Kommunikations- und Computertechnik den fast 150 000 Besuchern aus 51 Staaten zu demonstrieren und anzubieten. Diese Leistungsschau wurde von einem außerordentlich anspruchsvollen Tagungs- und Kongreßprogramm begleitet, das fast 4600 Besucher, meist partiell, absolvierten.
Mit dieser Leistungsbilanz bestätigte die Systems '87 ihren Anspruch als größte internationale Fachveranstaltung für Computer und Kommunikation. Übersichtliche Gliederung nach Angebotsschwerpunkten und strenge Produktbeschränkung entsprechend der Zielstellung fanden einen guten Anklang bei den Besuchern. Besonders stark frequentiert waren die Angebotsbereiche Kommunikationstechnik, Software, Dienstleistungen und Systeme für den Büroarbeitsplatz. Bemerkens- und betrachtenswert ist die zunehmende Integration von Kommunikations- und Computertechnik in die Klein- und Mittelbetriebe.
In Verbindung mit der Systems '87 fand vom 20. bis 21. Oktober der 2. Internationale Kongreß der Gesellschaft für Informatik (GI) zu wissensbasierten Systemen statt. In den Eröffnungsansprachen wurde der GI wiederholt bescheinigt, daß sie sich große Verdienste bei der Entwicklung der künstlichen Intelligenz erworben hat. Davon zeugte auch dieser 2. Internationale Kongreß, auf dem in 15 Fachsitzungen 37 Vorträge gehalten wurden. Schwerpunkte dabei waren theoretische Grundlagen (z. B. Wissenspräsentation und automatisches Beweisen), Vorstellung von Forschungsprojekten (z. B. Prolog-Arbeiten in Karlsruhe; EUREKA-Projekt PROTOS) und Einsatz von Expertensystemen (z. B. in der Produktion, in der Diagnose und Instandhaltung).
Festzustellen ist, daß eine große Anzahl von Firmen Expertensysteme ausstellte und vorführte. Bemerkenswert ist, daß, wegen der grundsätzlichen Verfügbarkeit von 32-Bit-Personalcomputern mit leistungsfähigen magnetischen und optischen Speichern im Gigabyte-Bereich, auch die Expertensysteme auf diesen Computern lauffähig gemacht worden sind. Texas Instruments, Nixdorf, Digital Equipment, Siemens, IBM u. a. bekannte und leistungsfähige Firmen hatten Expertensysteme in ihrem Messeangebot. Die Anwendungen unterscheiden sich nicht von dem allgemeinen Trend der gegenwärtigen Entwicklung. Überall ist eine gewisse

Ernüchterung zu verspüren, weil ganz einfach der große Erfolg bisher nicht in der Breite eingetreten ist. Die Programme von Expertensystemen sollen nützlich, benutzbar, pädagogisch und leicht modifizierbar sein sowie Empfehlungen erklären, Fragen beantworten und neues Wissen erwerben können.

Eine erfolgreiche Realisierung eines Expertensystems hängt ganz wesentlich von der gestellten Aufgabe ab, die realisierbar, angemessen und bewertbar sein muß. Auch sollten die Grenzen des Expertensystems von vornherein formuliert werden, um Enttäuschungen zu vermeiden. Wichtig ist auch, eine kontinuierliche Systempflege zu betreiben. Einige Projekte sind wegen Nichtbeachtung obiger Faktoren gescheitert. Die Bedeutung der Schnittstelle Mensch-Computer wird zu häufig unterschätzt. Der Anwender ist völlig überfordert, wenn er sich erst in Prolog oder LISP einarbeiten muß. Zum Bedienen eines Expertensystems muß er kein KI-Experte sein. GI-Kongreß und Systems '87 lassen keinen Zweifel an dem weiteren Aufstieg und der zunehmenden Bedeutung und vorteilhaften Anwendung von Expertensystemen.

Auf dem Symposium zu „Trends in der Telekommunikation" wurde zur Telekommunikation in Europa, zum Strukturwandel in der Telekommunikation sowie über technische Möglichkeiten und praktische Realitäten der neuen Telekommunikation vorwiegend referiert.

Hervorzuheben ist auch das Symposium „Synergie von Optik und Elektronik", auf dem Probleme der optischen Schrifterkennung, der optoelektronischen Speichertechnik, der Umwälzung in der Drucktechnik sowie der Zukunftsperspektiven optischer Computer behandelt wurden. Den Ausführungen von *Laws* (Universität Duisburg) folgend, könnte in der zweiten Hälfte der 90er Jahre der optische Computer für Spezialaufgaben zur Verfügung stehen.

Auf der Messe nahmen die Bürokommunikationssysteme einen speziellen Ausstellungsbereich ein, wodurch ihre große Bedeutung unterstrichen wurde. Direkt neben dem Ausstellungsbereich „Kommunikationstechnik" angeordnet, der vom Telefon bis zum Glasfasernetz alles anbot, ergaben sich gute Informationsmöglichkeiten. Generell läßt sich feststellen, daß zur Grundausstattung eines Bürosystems ein Personalcomputer (meist 16-Bit-PC) mit Drucker und ein Computernetz gehören. Umfangreiche Softwarepakete gestalten die Grundkonfiguration zu einem leistungsfähigen Arbeitsmittel.

Bei den Personalcomputern war das breite Angebot an 32-Bit-Computern mit dem Mikroprozessor Intel 80386 besonders auffallend. Jede größere Firma bot einen solchen PC an. Ansonsten dominierte der 16-Bit-PC mit dem Intel-Prozessor 80286, häufig auch noch mit dem Koprozessor 80287 ausgerüstet. Der Hauptspeicher liegt meist bei 1 MByte. 2 Diskettenlaufwerke der Größe 5¼" und je 1 MByte Speicherkapazität sowie 1 Festplattenlaufwerk mit 20 ... 40 MByte gehören ebenfalls zum Standard.

Am vielfältigsten gestaltete sich das Angebot an Druckern. Nadel-, Thermo-, Tintenstrahl- und Laserdrucker waren vertreten. Der Laserdrucker gehörte durchweg zum Standardangebot. Er liegt allerdings noch über dem Preis der Nadeldrucker, die gegenwärtig am stärksten abgesetzt werden. Ein 24-Nadel-Drucker leistet in der Regel 100 Zeichen/s bei Schönschrift und 300 Zeichen/s bei EDV-Qualität; Hochgeschwindigkeitsdrucker ermöglichen den Druck von fast 500 Zeichen/s. Die Laserdrucker schaffen meist um 10 Seiten pro Minute als Tischdrucker, Spitzenmodelle leisten 18 Seiten/Minute. Mit den Videoprintern (Tintenstrahl oder Thermotransfertechnologie) lassen sich für hochauflösende CAD/CAM-Anwendungen farbige Hardcopies anfertigen.

Groß und variabel war das Angebot an externen Speichern. Hervorzuheben sind die 3,5"-Floppy-Disk-Laufwerke mit einer Speicherkapazität (unformatiert) von 1 MByte sowie einer Spur-zu-Spur-Positionierzeit von 3 ms. Die Compact-Disk (CD-ROM) hat sich überall eingeführt und wird als Massenspeicher zentraler Daten verwendet. Bei den Festplattenspeichern ist mit der 40-MByte-5¼"-Platte ein gewisser Standard erreicht. Bei den 5¼"-Diskettenlaufwerken sind bei der halben Bauhöhe bereits 6,6 MByte unformatiert auf einer Seite erreicht.

Die optischen Speicher existieren in der Compact-Disk sowie im WORM-Speicher (WORM = WRITE Once-Read Multiple). WORM-Speicher bieten im 5¼"-Format 600 MByte und im 12"-Format 2,6 GByte.

Eine neue Anwendungsform von PCs verbirgt sich hinter Desktop Publishing (DTP), was soviel wie „Publizieren vom Schreibtisch aus" bedeutet (s. auch MP 4/88). Die Grundvoraussetzung für DTP-Anwendungen ist eine leistungsfähige Textverarbeitung, mit der es auch möglich ist, beliebige Grafiken in den Text einzubauen. Diesem Leistungsanspruch genügen die meisten auf dem Markt befindlichen Softwarepakete für die Textverarbeitung noch nicht. Um DTP mit PCs zu betreiben, wird neben einer leistungsfähigen Textverarbeitungssoftware und einer flexiblen Grafiksoftware auch ein diesem Prozeß angepaßtes Layoutprogrammpaket benötigt. Mit DTP wird eine neue Qualität in der Bürokommunikation erreicht. Es sind 16- und 32-Bit-Systeme im Einsatz. Da meist eine längere Zeit am DTP-Platz gearbeitet wird, hat man zunehmend größeren Wert auf die ergonomische Gestaltung des Arbeitsplatzes gelegt. Als Eingabemedium dienen Abtastgeräte (sog. Scanner), Tastatur und Maus. Ganzseitenbildschirme höherer Auflösung bieten Übersicht und bequemes Arbeiten. Die Ausgabe erfolgt fast ausschließlich über Laserdrucker. Da diese DTP-Systeme sehr anwenderfreundlich in Bedienung und Nutzung und in jedem Büro und Labor einsetzbar sind, wird ihnen ein riesiger Marktanteil eingeräumt.

Währenddessen bei der Gerätetechnik eine außerordentlich dichte Warendecke vorhanden ist, existiert ein hoher Bedarf an Softwareprodukten, um die Gerätetechnik umfassend einzusetzen. Allerdings muß die Software so entwickelt sein, daß sie auf Computern des internationalen Standards lauffähig ist. Bei den bekannten Programmiersprachen ist der Trend zur Beschleunigung der Abarbeitung deutlich zu erkennen, da neben Turbo-Pascal auch Turboversionen von BASIC, C u. a. Sprachen angeboten wurden. Bei den Betriebssystemen stehen UNIX, CP/M und MS-DOS im Vordergrund.

Das Anwendersoftwareangebot auf der Systems '87 konzentrierte sich auf die Nutzung in Klein- und Mittelbetrieben sowie auf einen hohen Allgemeinheitsgrad. So präsentierten sich Programmpakete für die Verwaltung, wie z. B. Postvertrieb mit PC oder die Lagerverwaltung, das Abrechnungswesen, die Kalkulation, die Steuerberatung.

Der Verbund der Computer über lokale und globale Netze entwickelt sich mit einer beachtlichen Dynamik. Das wurde besonders unterstrichen durch die Demonstration von 36 Unternehmen im Rahmen eines sog. SYSTEMS-MultiNet auf der Systems'87. Die Zusammenschaltung unterschiedlicher Computer der verschiedenen Unternehmen zeigte den realen Stand der Kommunikationstechnik. An einem gemeinsamen Netz mit Terminals, PCs, Minicomputern und Großrechnern konnte jeder mit jedem kommunizieren. Als Übertragungsmedium dienten verdrillte Kupferadern sowie Koaxial- und Glasfaserkabel.

Mit dieser Demonstration sollte bewiesen werden, daß eine Netzkopplung ganz unterschiedlicher Systeme möglich ist, wobei kein Anbieter auf sein Betriebssystem zu verzichten brauchte.

Als Spitzenprodukt auf dem Softwaregebiet ist das 80386-Betriebssystem Microsoft Windows/386 von Microsoft zu nennen, das den Übergang zum angekündigten Betriebssystem Microsoft OS/2 sehr günstig gestalten soll.

Spitzenprodukte auf dem Hardwaresektor sind vor allem die 32-Bit-PCs von IBM, Siemens, Commodore, Nixdorf, Toshiba, Olivetti u. a. Herausragend ist der T 5100 von Toshiba, der mit dem I80386 ausgestattet ist, 4 MByte Hauptspeicher und eine 40-MByte-Festplatte hat und einschließlich Plasmabildschirm und Tastatur nur 6,8 kg wiegt. Als Spitzenleistung ist auch die Diskette von Verbatim mit 6,6 bzw. 12 MByte einzuordnen. Der kleinste PC-Drucker DICONIX 150 leistet 150 Zeichen/s, hat die Abmessungen 27 × 16 × 5 cm³ und wiegt keine 2 Kilogramm. Schließlich gehören auch die optischen Speicher (WORM) zu den herausragenden Ausstellungsprodukten.

Prof. Dr. Walter Cimander

Expertensysteme '87

Am 3. und 4. Dezember 1987 fand in Suhl das erste nationale KDT-Kolloquium zum Rahmenthema *Expertensysteme* statt. Es wurde gemeinsam von der Wissenschaftlichen Sektion „Computer- und Mikroprozessortechnik" (Fachverband Elektrotechnik), dem Fachausschuß *Beratungs- und Expertensysteme* (Wissenschaftliche Gesellschaft für Meß- und Automatisierungstechnik) und der Fachsektion *Künstliche Intelligenz* Gesellschaft für Informatik der DDR) veranstaltet. Das Ziel der künftig im zweijährigen Rhythmus in Suhl stattfindenden Informationskolloquien besteht vor allem in der Popularisierung von Theorie und Praxis des Entwurfes und der Nutzung von elementarintelligenten Computern, deren Hauptmerkmal die Fähigkeit zur logischen **und** algorithmischen Verarbeitung von deklarativem und prozeduralem Wissen ist. Damit werden wesentliche Leistungsmerkmale der 5. Rechnergeneration erfüllt.

In den Vorträgen zum Kolloquium 1987 wurden Grundlagenwissen vermittelt, Entwicklungstendenzen aufgezeigt und vor allem Beispiele der Anwendungen von Beratungs- und Expertensystemen in der DDR erläutert. Mit etwa 180 Teilnehmern wurde ein angemessener Kreis von vorwiegend Ingenieuren, wissenschaftlichen Mitarbeitern und talentierten Studenten erreicht.

Die Grundlagenvorträge überstrichen aktuelle Themen wie „methodische und rechentechnische Grundlagen von Expertensystemen" *(P. Florath, ZKI der AdW Berlin)*, „unscharfe Logiken für Expertensysteme" *(Ch. Posthoff, TU Karl-Marx-Stadt)* und „Programmieren in PROLOG" *(R. Knauf, TH Ilmenau)*. Prinzipielle Aspekte und perspektivische Orientierungen wurden mit den Themen „Computer im Jahr 2000 – Expertensysteme" *(M. Roth, TH Ilmenau)* und „Beratungs- und Expertensysteme in automatisierten Prozessen" *(J. Wernstedt, TH Ilmenau)* gegeben. In einem Vergleich prozeduraler und deklarativer Sprachen wurden deren Besonderheiten für den Bau von Expertensystemen herausgearbeitet *(H. Killenberg, TH Ilmenau)*. Sehr instruktive Anwendungsprojekte von Expertensystemen wurden mit „PROCON I – ein KI-Werkzeug zur Prozeßautomatisierung" *(B. Boehme)*, einem Beratungssystem zur „Konfigurierung von Robotron aus Moduln" *(W. Holle, TH Ilmenau)* zur „Wälzlagerauswahl" *(H. Wolf, SW Karl-Marx-Stadt)* zur „Entscheidungsfindung im Schiffsführungsprozeß" *(V. Koehler, SE Rostock; K.-D. Weigelt, IHS Warnemünde)* dargestellt. Einen Höhepunkt bildete der Vortrag „Das Expertensystem DIDIMA zur Unterstützung der Frühdiagnostik von Brustdrüsenkrebs" *(W. Schueler, Bezirkskrankenhaus Karl-Marx-Stadt)*.

Prof. Dr. Dr. Michael Roth



Arbeiten an der Theorie der Hochtemperatur-supraleitung

Die Erfolge auf dem Gebiet der Supraleitung in technisch einfacher beherrschbaren Temperaturbereichen haben die Wissenschaftler zur Erarbeitung einer Theorie dieser Erscheinungen herausgefordert.
Im Institut Laue-Laugevin in Grenoble wurde mit Hilfe der hochauflösenden Pulverdiffraktometrie die Struktur des 100-Kelvin-Supraleiters $YBa_2Cu_3O_7$ im Detail geklärt. Von den französischen Forschern konnten die zuvor mit anderen Verfahren gefundene Schichtstruktur des Supraleiters bestätigt und darüber hinaus weitere Fragen der Erscheinung geklärt werden. Die neuen Materialien bestehen aus einzelnen durch Barium- und Yttrium-Atomen getrennten Schichten aus Kupfer- und Sauerstoffatomen in abwechselnd zweidimensional oder linear vernetzter Form.
Erstmals konnte in den verschiedenen Schichten der Bindungszustand der Kupferatome gemessen werden. Es sind sowohl dreiwertige als auch zweiwertige Kupferatome in beiden Ebenen gleichzeitig vorhanden. Die räumliche Fluktuation des Bindungszustandes innerhalb der Ebene könnte für das Auftreten elektrischer Leitung in diesen normalerweise elektrisch isolierenden Keramiken verantwortlich sein.
Offen ist noch, ob die gemessene Transversalschwingung großer Amplitude in den linearen Kupfer- und Sauerstoffatomen zur Supraleitung beiträgt. Ebenso konnte noch nicht ermittelt werden, ob die linearen oder flächigen Strukturen für die Supraleitung ausschlaggebend sind.
Die Hypothese, daß die Supraleitung durch eine magnetische Wechselwirkung zwischen den Elektronenpaaren möglich wird, konnte bei YBa_2Cu_3O7 nicht bestätigt werden.
Obwohl die Theorie der Supraleitung bei hohen Temperaturen noch nicht vorliegt, wird erwartet, daß die Pulverdiffraktometrie mit Neutronen zu einer Klärung der Vorgänge beitragen wird.

Wi

aus Blick durch die Wirtschaft 10. 11. 87

Neues portables Betriebssystem

In den USA und Großbritannien soll sich das portable Betriebssystem Pick neben Unix aufgrund seiner Ausrichtung auf kommerzielle Anwendungen, seiner Benutzerfreundlichkeit und dem Vorhandensein nur einer Grundversion zunehmender Beliebtheit erfreuen. Zu den Schwächen sollen die Kommunikationsfähigkeit und die Multitaskingmöglichkeiten gehören. Von einigen Firmen wird deshalb versucht, Pick und Unix nebeneinander laufen zu lassen. Dies soll durch eine menüunterstützte Benutzerschnittstelle zu Pick und Unix oder durch die Überlagerung des Unix-Kerns durch Pick erfolgen.

Wi

aus Diebold Management Report 6/1987

Standardisierungs-bestrebungen für Unix

Um der Unix-Vielfalt eine einheitliche Linie zu geben, soll sich eine Arbeitsgruppe der IEEE-Organisation um die Formulierung eines allgemeinen Unix-Standards bemühen. Mit Rücksicht auf mögliche Copyright-Verwicklungen soll der Standard die Bezeichnung Posix (Portable Operating System Interface) erhalten.
Bis Ende 1987 sollten die Schnittstellen zur Programmbibliothek, die Befehlssprachen und die Hilfsprogramme der „Shell" fertiggestellt werden. Der Standard soll der ISO vorgelegt werden, damit er verbindlich als Weltstandard erklärt wird. Skeptiker befürchten, daß bis zu diesem Zeitpunkt der Fortschritt das Problem bereits überholt haben könnte.

Wi

aus Diebold Management Report 6/87

PC-Programm für 3D-Bildverarbeitung

Dreidimensionale Bildverarbeitung soll mit Hilfe der PC3D-Software der Firma Jandel Scientific nunmehr vergleichsweise preiswert realisierbar sein. Aus Daten, die durch CAT, Scanner oder Fotos aus Elektronenmikroskopen gewonnen werden, sollen sich dreidimensionale Bilder auf IBM-PC oder kompatiblen PC darstellen lassen. Das auf dem Bildschirm erzeugte Bild soll gedreht oder zerteilt werden können.

Wi

aus Diebold Management Report 7/1987

Ausdehnung der Mailbox-Nutzung

Die GeoMail-Vereinigung, in der die Mailboxbetreiber der BRD (Mailbox GmbH, GeoNet Services, Infex) zusammengeschlossen sind, will bis zum Jahre 1990 rund 70000 elektronische Postfächer vermietet haben. Ihre Prognose gründet die Vereinigung auf die international wachsende Anzahl der Betreiber und Nutzer, die kommende Vernetzung aller Mailboxsysteme über die X.400-Protokolle und auf den erwarteten Erfolg eigener Marketingaktionen. Die GeoMail-Firmen wollen ihr Angebot gegenüber der Bundespost der BRD attraktiver gestalten. Dazu sollen der Übergang in das Telexnetz und in das Bildschirmtextsystem gehören. Den Mailbox-Teilnehmern würde damit der Zugang zu 1,5 Mio. Telexteilnehmern, 1 Mio. Fernkopierern und etwa 100000 Bildschirmtextteilnehmern über ein und dasselbe Terminal möglich.
Ein anderer Dienst soll Recherchen in den angeschlossenen Datenbanken ermöglichen, ohne daß es separater Verträge zwischen Host und Benutzer bedarf. Weitere Spezialdienste umfassen Sprachübersetzung, Desktop Publishing und einen zeitgesteuerten Anrufdienst.

Wi

aus Diebold Management Report 6/1987

Alternative zum IBM-Betriebssysten MVS durch KeyKOS

Mit finanzieller Unterstützung von Amdahl soll die Firma Key Logic (20 Beschäftigte) ein Betriebssystem erarbeitet haben, das leistungsfähiger und billiger als MVS sein soll.
Der Umfang der Programmzeilen wurde gegenüber MVS von 20 Mill. Zeilen auf etwa 500000 reduziert.
KeyKOS soll sich besonders für Online-Transaktionsverarbeitung, die Verwaltung großer Hauptspeicher im Gigabyte-Bereich, die Steuerung der Ein- und Ausgabetransaktionen von und zum Plattenspeicher eignen sowie über einen beständigen virtuellen Speicher, volle Fehlertoleranz und Objektprogrammierung verfügen. Die Programmiersprachen des IBM-Systems 370 werden unterstützt.
KeyKOS soll sich seit 9 Monaten im Test befinden. Die Firma Unisys soll eine Dauerlizenz für das Betriebssystem erworben haben.

Wi

aus Diebold Management Report 7/1987

Grenzen der Miniaturisierung?

Aufgrund von Untersuchungsergebnissen an Indium-Teilchen mit Abmessungen von einem Mikrometer sollen Wissenschaftler zu der Erkenntnis gekommen sein, daß Metallteilchen desto schlechter elektrischen Strom leiten, je kleiner sie werden. Abgemagert auf unter einen Zehntel Mikrometer wird ursprüngliches metallisches Material danach zu einem Nichtleiter. Von den vorstehend genannten Wissenschaftlern wurde für diesen Effekt der Begriff „Size-Induced-Metal-Insulator-Transition" (SIMIT) geprägt. Der größeninduzierte Metall-Isolator-Übergang läßt sich nicht mehr mit der klassischen Theorie erklären, welche die Ladungsträger, die für die Leitfähigkeit eines Materials verantwortlich sind, als geladene Kügelchen auffassen. Mit Hilfe der Quantenmechanik, die auch den Elektronen Welleneigenschaften zuspricht, läßt sich das Ergebnis verstehen. Die kleinen Halbleiter- oder Metallstückchen, bei denen der Effekt auftritt, werden deshalb auch „quantum dots" genannt.
Die Untersuchungen an Mikrokristallen, deren Durchmesser unterhalb eines Mikrometers liegt, werden derzeit im Zusammenhang mit der Herstellung immer leistungsfähigerer Rechnerchips weltweit intensiv betrieben.
Die Grenze für die Miniaturisierung würde nach den Ergebnissen der Wissenschaftler bei einem halben bis einem zehntel Mikrometer liegen.
Andererseits ermöglicht der gefundene Effekt die Miniaturisierung anderer Bauelemente, z. B. könnte man einen Widerstand dadurch erhöhen, indem man ihn physisch verkleinert, statt wie bisher, einen Metallstreifen verlängert.
Mit Hilfe der Mikrokristalle könnten sich auch neue künstliche Stoffe mit „maßgeschneiderten" elektrischen und magnetischen Eigenschaften herstellen lassen. So sollen sich neue Möglichkeiten auch für die Herstellung von Kondensatoren und Transformatoren ergeben.

Wi

aus Bild der Wissenschaft 12/1987

Touch-Screen mit einer einzigen LED

Einen neuartigen „Touch-Screen", der nur eine einzige Leuchtdiode benötigt, die ihr Licht quer über die Schirmfläche sendet, stellte die Firma Wells-Gardner Electronics Corp. in den USA vor. Das Licht wird quer über die Schirmfläche gesendet, trifft an der Gegenseite auf eine reflektierende Schicht und kehrt an die Ausgangsseite zurück. Dort trifft das Signal auf einen linearen CCD-Detektor (über eine Sammellinse). Das CCD-Element liefert eine Winkelinformation des reflektierten LED-Lichtes und vergleicht diese mit einem Durchschnittswert. Abweichungen zeigen an, daß das Licht an einer definierten Stelle des Schirms (durch Berührung) unterbrochen wurde. An der unteren Kante des Schirms befindet sich eine zusätzliche Spiegel-Leiste, über die ebenfalls ein (sekundäres) Reflexionssignal auf die CCD-Sensoren gelangt. Mit den beiden rückkehrenden Lichtsignalen gemeinsam läßt sich durch Triangulation der XY-Wert des Berührungspunktes errechnen. Das Auflösungsvermögen ist durch die Größe des CCD-Arrays gegeben. Es reicht derzeit für eine Schirmgröße von 4 × 9 Zoll aus.

Wi

aus Elektronik 21/1987

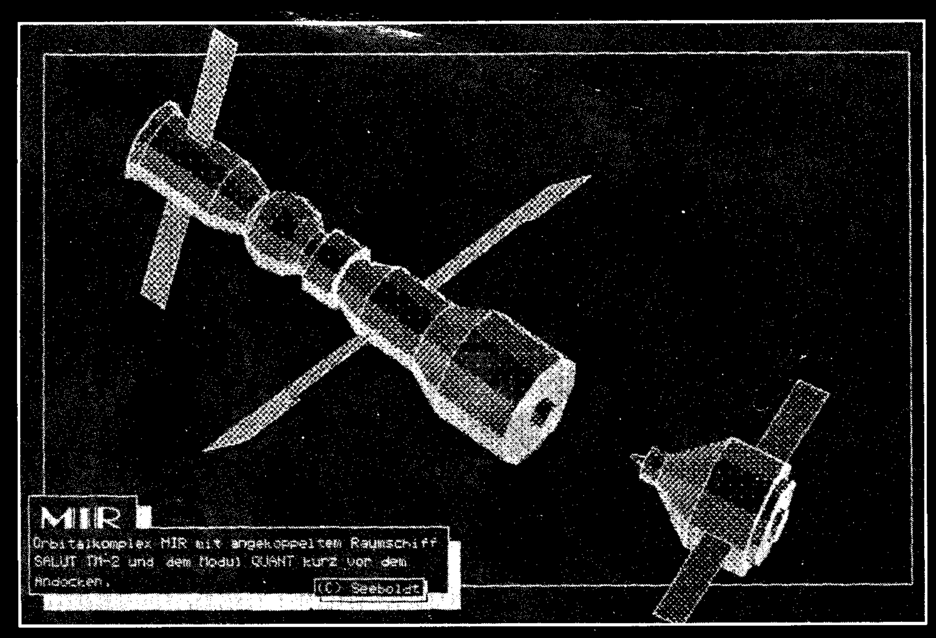
Dokumente elektronisch archiviert

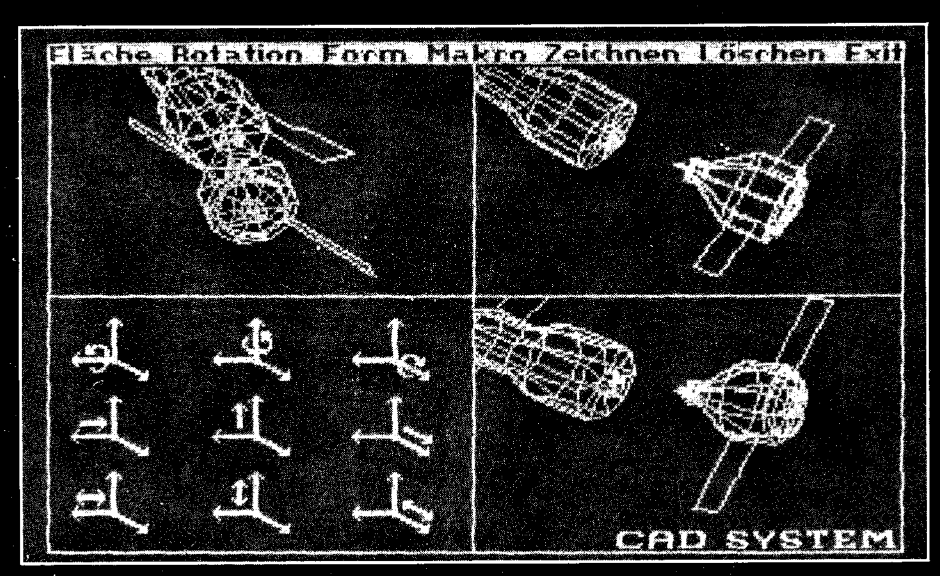
Die Firmen Janisch & Klass-Computertechnik und Minox entwickelten ein Dokumenten-Archivierungs-Subsystem mit der Bezeichnung „Minidok". Das System verfügt über eine V.24-Schnittstelle und ist an jeden Hostrechner anschließbar. Das Retrieval-System steuert die Dokumenten-Ein-/Ausgabe mit Hilfe von ASCII-Befehlen.
An das Subsystem lassen sich entweder Telefaxgeräte oder Scanner und Laserdrucker anschließen. Der Bildschirm hat eine Auflösung von 824 × 1024 Bildpunkten.
Die Kapazität des Datenspeichers umfaßt 800 MByte (16000 Seiten). Es lassen sich auch CD-ROMs verwenden.
Aufgrund der Verwendung von Telefaxgeräten werden die Informationen im CCITT-Format gespeichert und so lassen sich Dokumente über das Fernmeldenetz senden und empfangen.
Das System kann in allen Bereichen eingesetzt werden, in denen Dokumente (Zeichnungen, Pläne, Schriftstücke usw.) archiviert werden müssen.

Wi

aus Elektronik 21/1987

Beispiel für Grafikmöglichkeiten auf einem 8-Bit-Computer





Die Raumstation MIR wurde auf einem 8-Bit-Heimcomputer konstruiert. Dabei kam ein leistungsfähiges CAD-Programm zum Einsatz, das in Overlaytechnik arbeitet, um den zur Verfügung stehenden Speicherplatz von 64 K Byte optimal zu nutzen. Es versteht sich von selbst, daß die Rechenroutinen alle in Assembler programmiert sind.

Zuerst wurde ein Drahtgittermodell aus einzelnen Makros im Editor erstellt. Die Makros können mehrfach aufgerufen und beliebig verformt werden. Zum Schluß der Konstruktionsphase bestand das Objekt aus drei großen Makros: MIR, SOJUS und QUANT. Diese wurden dann entsprechend zusammengefügt und in die jetzige Position gedreht. In einer speziellen abschließenden Rechenarbeit erzeugte der Computer die vorliegende schattierte Variante mit einer angenommenen Beleuchtungsquelle von vorn oben. Dabei wurde vom Rechner eine leicht perspektivische Sicht auf die Station erzeugt. Diese Rechenroutine lieferte gleichzeitig die erhöhte Darstellung des CAD-Objektes in 640 mal 400 Bildpunkten, was eine akzeptable Grafik liefert. Man könnte das Gebilde auch als Körper ohne Schattierung berechnen lassen.

Zusätzlich habe ich vom Computer noch einen Film zu 24 Bildern einer kompletten Drehung der Raumstation um die x-Achse berechnen lassen. Leider läßt sich diese Arbeit hier schlecht demonstrieren. Das gesamte Werk benötigte immerhin 11 Stunden Rechenzeit, was somit auch auf einen Hauptmangel von 8-Bit-CAD-Systemen hinweist, die extreme Langsamkeit der Abarbeitung. Andererseits ist es eine sehr gute Demonstration der Fähigkeiten von kleinen Computern mit wenig Speicherplatz, wenn optimale Software genutzt wird.

S. Seeboldt
Computerklub im Berliner Haus der Jungen Talente

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 5
S. 129–160



ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPU[illegible]CHNIK · INFORMATIK

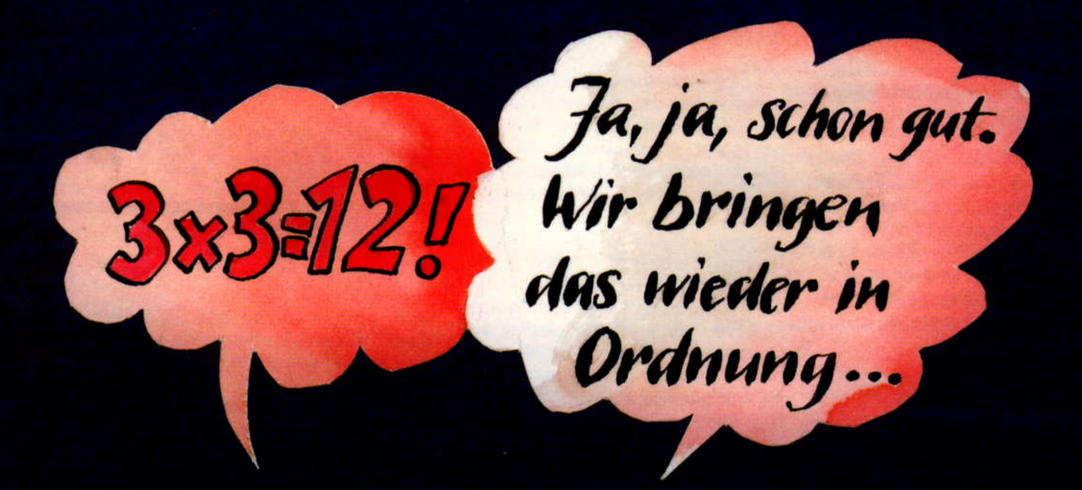
MP

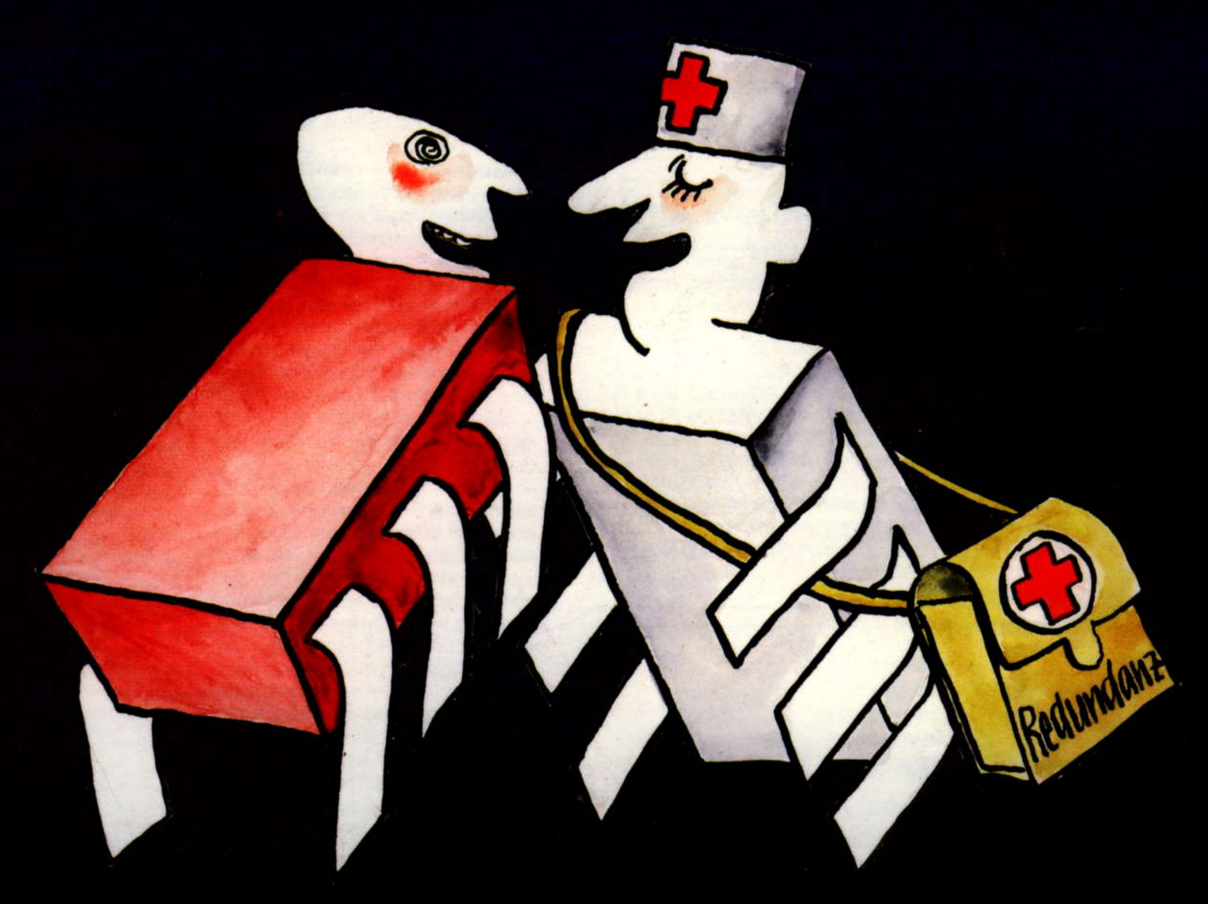
Heft 6 · 1988

Mikroprozessortechnik

VEB Verlag Technik Berlin

ISSN 0232-2892





Fehlertolerante Systeme

Prüfsummen am BC/PC

Bemerkungen zum Leserbrief in MP 9/87, S. 258

„Veröffentlichung von Programmen im DUMP-Format" von Dr. V. Richter

Ich vertrete auch die Meinung, daß man bei Veröffentlichungen von Programmcode, ob nun im DUMP- oder im Quelltext-Format, unbedingt auf Faksimile orientieren sollte. Das ist ja auch als Ihre Meinung aus der redaktionellen Stellungnahme zu oben genanntem Leserbrief herauszulesen.
Was nun die Prüfsummen oder ähnliches angeht, könnte man bei Texten, die als Diskettendatei zur Faksimile-Verwendung ausgedruckt an die Redaktion eingereicht werden, am Schluß des Artikels eine Angabe zu diesen Bildern machen, die folgendes Aussehen hat:

Anmerkung:
Checksumme mit POWER CHECK berechnet.
Bild 1 – 4C61 dec: 19553
Bild 2 – 71E0 dec: 29152
Bild 3 – E473 dec: 58483
Bild 4 – F898 dec: 63640

Für all die Fälle, wo sowohl beim Autor als auch beim Leser das Dienstprogramm POWER (DIENST vom VEK Robotron) verfügbar ist, also z. B. für BC/PC der 8-Bit-Technik, könnte der Leser nach dem Eintippen durch Gegenvergleich sich und die Richtigkeit der Veröffentlichung überprüfen bzw. fundiert an den Autor herantreten. Die Angabe der Checksumme hexadezimal und dezimal dient der höheren Sicherheit bezüglich Druckfehlererkennung durch den Leser.
Zur Demonstration teile ich Ihnen folgendes Bild 1 mit, das eine TURBO-PASCAL-INLINE-Prozedur zum dynamischen Aufruf eines CP/M-Kommandos einschließlich Parameterangaben aus einem TURBO-Programm heraus analog zum dBASE II-Kommando: „QUIT TO '⟨name⟩ [⟨parm⟩]'" enthält.
Diese wenigen Programmzeilen kann ein interessierter Leser leicht abtippen. Im Bild 2 ist der Aufruf zur Checksummenbildung und ein Testbeispiel für die mitgeteilte INLINE-Routine gezeigt.
Das Bild 1 ohne den Rahmen, d. h. nur der Text, den ein interessierter Leser abtippen würde, kommt aus einer Diskettendatei: PGMRUF. INC.

Anmerkung: Checksumme mit POWER CHECK berechnet.
Bild 1 – B482 dec: 46210

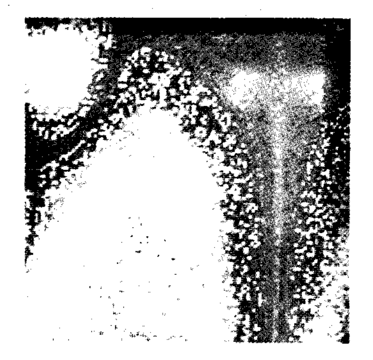
Christian Hanisch, Berlin

```
PROCEDURE PGMRUF(VAR PGMNAM:Str32);
{Mit PGMRUF(PGMNAM) wird ein CP/M-Kommando analog zu:
QUIT TO "<name>[ <parm>]"
aufgerufen. Das CP/M-Kommando, z.B. "A:WS   " muss in der
Stringvariablen PGMNAM (max. 32 Zeichen) enthalten sein.}
begin inline
(¤2A/PGMNAM/¤7E/¤A7/¤FE/¤00/¤28/¤1E/¤FE/¤20/¤30/¤1A/¤4F/
¤32/*+77/¤06/¤00/¤11/*+73/¤23/¤ED/¤B0/¤EB/¤3E/¤24/¤23/¤77/
¤2B/¤3E/¤00/¤77/¤23/¤23/¤77/¤23/¤77/¤11/*+52/¤0E/¤1A/¤CD/
¤05/¤00/¤11/*+80/¤0E/¤13/¤CD/¤05/¤00/¤11/*+72/¤0E/¤16/¤CD/
¤05/¤00/¤FE/¤FF/¤CA/¤00/¤00/¤3E/¤00/¤32/*+89/¤11/*+54/¤0E/
¤15/¤CD/¤05/¤00/¤11/*+46/¤0E/¤10/¤CD/¤05/¤00/¤C3/¤00/¤00/
¤04/¤41/¤3A/¤57/¤53/¤00/¤24/¤00/¤00/¤00/¤00/¤00/¤00/¤00/
¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/
¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤01/¤24/¤24/¤24/¤20/¤20/
¤20/¤20/¤20/¤53/¤55/¤42/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/
¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/¤00/
¤00/¤00) end;
```

```
A>POWER C:<CR>
   ...
   ...
C=check pgmruf.inc<CR>

C:PGMRUF  .INC - checksum: B482  total: B482  dec: 46210
C=exit<CR>

A>TURBO<CR>             T e s t e n   der PGMRUF-Routine:
   ...
Program PGMRUF_Test;
TYPE Str32=STRING[32];
Var PGMNAM:Str32;
{¤I PGMRUF.INC }
BEGIN
PGMNAM:='A:STAT B:*.*';
PGMRUF(PGMNAM)
END.
>run
```




Chaosgrafik

Ich beschäftigte mich seit 2 Jahren mit der Erzeugung von „Mandelbroten", „Feigenbäumen" und „Juliagrafiken", daher einige Anmerkungen zu den Beiträgen von Professor Horst Völz in MP 1/88.
In /1/ und /2/ sind neben sehr vielen schönen Bildern („Spiralnebeln", „Polypen", „Seepferdchen" usw.) auch die theoretischen Grundlagen der Fraktale dargelegt. Im März 1980 fand Benoit B. Mandelbrot seine berühmte Figur, der die Bremer Forschungsgruppe für komplexe Dynamik später den Namen „Apfelmännchen" gab. Mandelbrot entwickelte den Begriff „Fractal" für seine neue Kategorie von geometrischen Figuren, die sich nicht mehr mit den herkömmlichen Begriffen – Punkt – Linie – Fläche – Körper – beschreiben lassen. Die Grenze des Apfelmännchens ist weder Linie noch Fläche, sie ist unendlich lang von der Breite 0. Ordnung geht über in Chaos (deshalb spricht man auch von „Chaosgrafik") und umgekehrt. Fraktale sind durch zwei Eigenschaften gekennzeichnet:
– die Selbstähnlichkeit; in jedem Ausschnitt finden wir die Form des Ganzen wieder
– es treten keine exakt begrenzten Flächen auf; eine Länge oder ein Flächeninhalt ist nicht bestimmbar.
Mandelbrot untersuchte die Funktion $y = x^{z+1}$ in der komplexen Zahlenebene. (Jede komplexe Zahl kann als Punkt mit x; y;-Koordinaten aufgefaßt werden.) Die Iterationsgleichung, die zur Erzeugung des Apfelmännchens führt, ist $z_{n+1} = z_n^2 + c$. Überträgt man diese Gleichung ins Komplexe, so ergibt sich $z = x + y_i$ und $c = c_r + c_i$ oder $f(z_n) = {x_{n-1}}^2 - {y_{n-1}}^2 + c_r + i(2x_{n-1}y_{n-1} + c_i)$. Es können 4 Parameter verändert werden: x, y, c_r, c_i. Bei der Generierung des Apfelmännchens wird die Iteration mit $x = y = 0$ begonnen, c_r und c_i werden für jeden Bildpunkt verändert. Hält man c_r und c_i für ein Bild konstant und verändert x und y, so erhält man Bilder von Julia-Mengen (Gaston Julia beschäftigte sich bereits 1920 mit der Iteration von komplexen Funktionen, ihm zu Ehren werden die fraktalen Grenzen dieser Bilder heute Julia-Mengen genannt).
Nun noch ein Tip zur Beschleunigung der Berechnung eines Apfelmännchens, der in /3/ beschrieben wird. Der schwarze Körper ist eine Kardioide mit $r_1 = 1/4$ und $r_2 = 1/4$. Der Kopf ist ein Kreis mit Mittelpunkt 1; 0; und Radius 1/4. Mit folgendem Test wird die Berechnung in diesen Teilen des Apfelmännchens, die bekanntlich am längsten dauert, unterdrückt.

IF CR > 0.75 THEN Test auf Apfelmännchenkörper
ELSE Test auf Apfelmännchenkopf
Test auf Apfelmännchenkörper
Y2=YC*YC:X2=XC*XC
R=X2+Y2:S=SQR(R-XC*0.5+0.0625)
IF(16*R*S) > (5*S-4*XC+1) THEN ITERATION ELSE Zähler=MAX
Test auf Apfelmännchenkopf
Y2=YC*YC:X2=(XC+1)*(XC+1)
IF(X2+Y2) 0.0625 THEN ITERATION ELSE Zähler=MAX
Die Bilder zeigen Computergrafiken nach der Formel:
xk=cr yk=ci
Iteration
x=xk*cos(yk)
y=xk*sin(yk)
xk=x yk=y
Abbruchkriterium
x=xk oder Zähler=max
Dabei gelten für Bild 1:4 < cr < 5; 1 < ci < 1,8: max=16 und für
Bild 2: 4.2 < cr < 4,7; 1,6 < ci < 2; max=16.

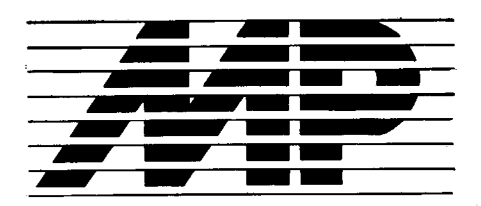
Literatur

/1/ Becker, K.-H.; Dörfler, M: Computergrafische Experimente mit Pascal (Ordnung und Chaos in dynamischen Systemen). Vieweg Verlag, Braunschweig, 1986
/2/ Peitgen, H.-O.; Richter, P. H.: The Beauty of Fractals. Springer Verlag, Berlin/Heidelberg/New York/Tokyo, 1986
/3/ Durandi, W.: Schnelle Apfelmännchen, c't magazin für computertechnik (1987) 3, S. 92–94

Brunhild Pflugk, Rostock

Bild 1: „QUIT TO ⟨name⟩[⟨parm⟩]" aus TURBO-Pascal heraus
Bild 2: Berechnen der Prüfsumme mittels POWER CHECK und Testbeispiel der INLINE-Routine PGMRUF

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 6 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 2 87 00, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2 87 03 71); Herbert Hemke, Redakteur (Tel. 2 87 02 03); Sekretariat Tel. 2 87 03 81

Gestaltung Christina Bauer

Titelgrafik H. J. Eggstein

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

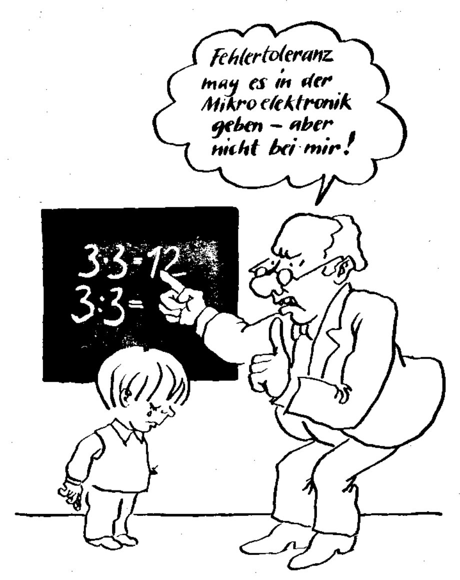
Redaktionsschluß: 19. April 1988

AN (EDV) 49837

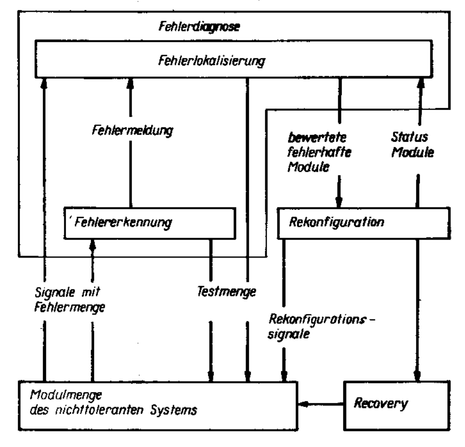
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

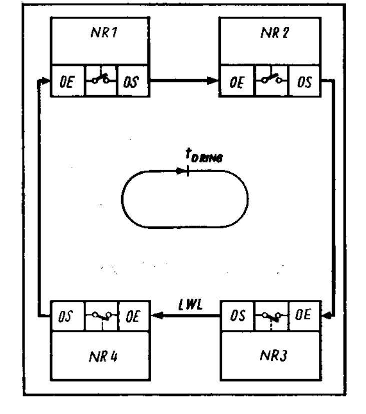
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tisku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Há Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



Seite 163



Seite 165



Seite 183

Inhalt

Erstes Rechnernetz für Plattenwerk



Foto: Weiß

Gegenwärtig wird im Plattenwerk Halle-Neustadt des Wohnungsbaukombinates Halle ein automatisiertes Leitungssystem aufgebaut. Damit wird erstmals in einer Vorfertigungsstätte des DDR-Wohnungsbaus eine lokale Vernetzung von Mikrorechnern angestrebt. Bisher gab es nur Lösungen mit Großrechneranlagen, die rund viermal so teuer sind. Das Vorhaben wird über drei Etappen bis 1991 realisiert. Zur Zeit werden rechnergestützte Einzelarbeitsplätze für die Produktionslenkung und -kontrolle, die komplexe Materialdisposition sowie die gesamte Planung und Ökonomie geschaffen. Durch drei 16-Bit-Rechner können in diesem Jahr bereits neun Arbeitskräfte, vor allem für die Qualitätskontrolle im Schichtbetrieb, gewonnen werden. Ab 1989 sieht die Konzeption den Einsatz des neuen Betriebsdatensystems BDS A 5230 vor, das auf der diesjährigen Leipziger Frühjahrsmesse erstmals vorgestellt wurde (siehe Bild). Kern des Systems ist eine Systemsteuereinheit – eine Mehrprozessorlösung mit zwei funktionsbestimmenden 16-Bit-Prozessoren (dem System-Erfassungs-Prozessor und dem System-Verarbeitungs-Prozessor), mit 2...4 Festplattenspeichern, einem oder zwei 1/2"-Magnetbandgeräten, einer Bedieneinheit (Display, Tastatur, Drukker) und der Anschlußmöglichkeit von bis zu 24 Bildschirmterminals über V.24 bzw. 20-mA-Stromschleife bzw. 60 Betriebsdatenterminals über Linieninterface IFLS-Z. Die Terminals sind RAM-ausgestattete Geräte, die von der Systemsteuereinheit je nach Anwendungszweck mit einer einsatzbezogenen Programmspezifizierung versorgt werden.
20 Datenterminals werden an das BDS in den Betriebsteilen Halle-Neustadt und Merseburg angeschlossen, um einen täglichen Informationsaustausch über alle wichtigen Produktionsdaten zu sichern. Schließlich sollen alle bestehenden Personalcomputer im Betrieb an das Datensammelsystem gekoppelt und miteinander vernetzt werden. *ADN/MP*

Jährlich 100 neue Schaltkreistypen

Mehr als 760 Grundtypen elektronischer Bauelemente gehören derzeit zum Fertigungsprogramm des Kombinates Mikroelektronik Erfurt. Seit seiner Gründung vor zehn Jahren hat sich das Sortiment leistungsfähiger Schaltkreise aus eigener Produktion vervierfacht. Die 59000 Beschäftigten in den 22 Kombinatsbetrieben leisteten durch die Entwicklung und Produktion immer höher integrierter Bauelemente einen gewichtigen Beitrag für die breite Einführung dieser Schlüsseltechnologie in der Volkswirtschaft in der DDR. Um für neue Schaltkreisgenerationen den erforderlichen wissenschaftlich-technischen Vorlauf zu sichern, arbeiten die Forscher und Ingenieure mit mehr als 30 Hoch- und Fachschulen sowie Institutionen der Akademie der Wissenschaften auf der Grundlage langfristiger Koordinierungs- und Leistungsverträge zusammen. Diese Kooperation trägt dazu bei, daß jährlich im Durchschnitt 80 bis 100 neue Typen in die Fertigung übergeführt werden können. Wichtigstes Vorhaben in diesem Jahr ist dabei die Produktion der 64-Kilobit-Speicherschaltkreise. Vorbereitet wird die Herstellung schneller 16- und 32-Bit-Mikroprozessoren.

ADN

Erste EC 1057 übergeben

Die erste EDVA EC 1057 ist Anfang März vom Kombinat Robotron an die künftigen Nutzer übergeben worden. Sie gehört zu einer neuen Gerätegeneration der ESER-Serie und kommt im VEB Datenverarbeitungszentrum Dresden zum Einsatz. Als Doppelprozessoranlage konzipiert, verfügt sie über eine Rechengeschwindigkeit von 1,6 Millionen Operationen in der Sekunde und über die annähernd dreifache Systemleistung ihrer Vorgängertypen. Die Kollektive im Anwenderbetrieb nehmen jetzt die Erprobung der Neuentwicklung im Dauerbetrieb unter Produktionsbedingungen vor. Nach entsprechender Einlaufzeit soll mit der EC 1057 künftig eine zentrale Datenbank der DDR-Elektronik geschaffen werden, mit der rund 200 Partner ständig im Dialog stehen können.

ADN

Konsultationsstützpunkt

Als erste Produktionsgenossenschaft im Bezirk Halle eröffnete die PGH Rundfunk Gernrode am 1. März einen Konsultationsstützpunkt zur Hard- und Softwareberatung für den PC 1715. Zum Service gehören die Unterstützung der Handwerksbetriebe bei der Auswahl des Rechners einschließlich der kompletten technischen Ausstattung sowie Hinweise für ökonomische Anwenderprogramme. Darüber hinaus werden Möglichkeiten zur Reparatur von Computern geschaffen. Mit diesem Konsultationsstützpunkt sollen Verwaltungsprozesse im Handwerk des Bezirkes sowie angrenzender Genossenschaften im benachbarten Bezirk Magdeburg weiter rationalisiert werden. Bei der Einrichtung des Stützpunktes arbeitete die PGH Gernrode eng mit der Ingenieurschule für Elektrotechnik und Maschinenbau Eisleben zusammen. Entsprechend einer wissenschaftlich-technischen Vereinbarung beider Partner halfen Eislebener Wissenschaftler und Studenten mit Software und Rationalisierungsprojekten.

ADN

informatica 88

Die erste internationale Konferenz und Messe *informatica 88* fand Mitte Februar in Havanna statt. Während des Treffens hatten mehr als 1200 Spezialisten aus rund 30 Ländern über neue Tendenzen auf dem Gebiet der Soft- und Hardware sowie über die Verstärkung der Kooperation beraten. Über 25000 Messebesucher machten sich mit neuen elektronischen Geräten und Ausrüstungen aus dem In- und Ausland, darunter mit 16-Bit-Technik des Kombinats Robotron, sowie deren Einsatz in verschiedenen Bereichen vertraut.

ADN

Singapur setzt noch stärker auf Elektronik

Die elektronische Industrie des südostasiatischen Stadtstaates war auch im vergangenen Jahr der am stärksten expandierende Zweig. Das geht aus einem jetzt veröffentlichten Wirtschaftsbericht 1987 hervor. Allein die Produktion von Diskettenlaufwerken für Computer stieg um 75 Prozent. Die in Singapur ansässigen 240 Unternehmen der Mikroelektronikindustrie mit zur Zeit mehr als 70000 Werktätigen stellen gegenwärtig fast alle Komponenten für Produkte der Heimelektronik, für Personalcomputer und deren periphere Geräte sowie Chips her. Die meisten Betriebe sind Firmen mit starker Beteiligung von Investoren aus Japan, den USA und Westeuropa, für die Singapur durch geringe Lohnkosten bei hohem technischen Niveau der Facharbeiter attraktiv ist. Im Rezessionsjahr 1985/86, als Singapur in einigen Wirtschaftszweigen einen Rückgang von fast zehn Prozent hatte, galt die Elektronikbranche als relativ „krisenfest". Auch künftig soll die Computerindustrie als wirtschaftliche Säule staatliche Förderung genießen.

ADN

Kuba steigert Elektronik-Produktion

Kubas Herstellerwerk von elektronischen Bauelementen wird seine Produktion in diesem Jahr gegenüber 1987 mehr als verdoppeln. Rund 20 Millionen elektronische Bauelemente – ein Viertel davon integrierte Schaltkreise – werden bis Jahresende das Halbleiterwerk „Che Guevara" in der westkubanischen Provinzstadt Pinar del Rio verlassen. Die Herstellung von Computerterminals steigt um das Dreifache. Im Rahmen des RGW hat sich Kuba auf Rechnertastaturen spezialisiert. Zu den größten Abnehmern kubanischer Elektronik gehört die UdSSR. Sie wird in diesem Jahr mit 8000 Bildschirmterminals dreimal mehr Geräte beziehen als 1987. Auch die DDR, Polen, Bulgarien und die ČSSR verwenden zunehmend Bauelemente „Made in Cuba".

ADN

TERMINE

Fachtagung Elektromagnetische Verträglichkeit in der Automatisierungstechnik

WER? Bezirksverband Dresden der KDT und Sektion Elektrotechnik der TU Dresden

WANN? 13. September 1988

WO? Hygienemuseum Dresden

WAS?
- EMV-Grundlagen
- EMV-gerechte Entwicklung und Konstruktion von Automatisierungsgeräten
- EMV-Störfestigkeitstestung von Funktionseinheiten und anderen Geräten
- EMV-gerechte Gestaltung von Automatisierungsanlagen
- EMV-gerechte Bauausführung von Erdungsanordnungen
- Maßnahmen gegen Blitz- und Abschaltüberspannungen sowie Stromrichternetzrückwirkungen
- Schutz gegen elektrostatische Entladungen

WIE? Anfragen an: KDT-BV Dresden, Basteistraße 5, Dresden, 8020 oder TU Dresden, Sektion Elektrotechnik, BAS, Mommsenstr. 13, Dresden, 8027

Habiger

Fehlertolerante Mikrorechnersysteme

Ingolf Fölsch,
Dr. Hans-Georg Porep,
Ronald Scheel
Ingenieurhochschule Wismar,
Sektion TdE/E

Mit dem Einsatz der modernen Mikrorechner in der Steuerungstechnik wachsen auch die Anforderungen an die Verfügbarkeit und Zuverlässigkeit solcher Systeme. In Fachartikeln, die sich mit den erhöhten Anforderungen an die Zuverlässigkeit solcher Mikrorechnersysteme befassen, rückt der Ausdruck Fehlertoleranz immer mehr in den Mittelpunkt des Interesses. Dieser Beitrag soll einen einführenden Überblick zum Fachgebiet der Fehlertoleranten Mikrorechnersysteme geben.

Die Entwicklung der Mikrorechentechnik war bislang vor allem durch das Bemühen um Leistungssteigerung gekennzeichnet. Erhöhung der Taktfrequenz, Verbreiterung der Datenformate, überlappte Befehlsverarbeitung und schließlich parallele Verarbeitung in Mehrprozessorsystemen sind Wege zur Leistungssteigerung. Komfortablere Betriebssysteme vereinfachen und erweitern die Anwendung. Damit wächst aber auch die Zahl der Fälle mit erhöhten Anforderungen an die Zuverlässigkeit. Der Ausdruck Fehlertoleranz rückt heute zunehemend in den Mittelpunkt des Interesses. Die vorliegende Arbeit soll helfen, auf diese Entwicklung aufmerksam zu machen und einen Überblick über Methoden, Möglichkeiten und Grenzen der Fehlertoleranz zu geben, soweit das aus heutiger Sicht möglich ist.

Zur Erreichnung einer hohen Zuverlässigkeit von digitalen Systemen gibt es zwei unterschiedliche Philosophien: Fehlervermeidung und Fehlertoleranz.

Kerngedanke der Fehlervermeidung ist es, die Fehlerursachen weitestgehend zu eliminieren, so daß ein weitestegehend perfektes System entsteht. Treten dennoch während des Betriebes Fehler auf, so führen diese zum Fehlverhalten des Systems.

Kerngedanke der Fehlertoleranz ist es, das System mit einer gewissen Redundanz zu versehen, die beim Auftreten von Fehlern wirksam wird, so daß diese zu keinem Fehlverhalten des Systems insgesamt führen, d. h. toleriert werden können.

Die Prinzipien zum Aufbau fehlertoleranter Systeme sind teilweise seit längerem bekannt, wurden aber in der Vergangenheit nur für spezielle Anwendungen realisiert. Beispiele sind zentrale Rechnersteuerungen für Nachrichtenvermittlungssysteme, die eine Nichtverfügbarkeit von weniger als einer Stunde in zehn Jahren erreichen, und Steuerungssysteme der Raumfahrt mit Missionszeiten von mehreren Jahren.

Der heutige Stand der Mikroelektronik erlaubt nun eine weitergehende Anwendung der Fehlertoleranz in der Rechentechnik. Das darf aber nicht nur als Anwendung bereits bekannter Prinzipien verstanden werden, sondern muß auch die Suche nach den neuen technologischen Möglichkeiten angepaßten Lösungen bedeuten.

Fehler

Die Struktur fehlertoleranter Systeme wird wesentlich von der Art der zu tolerierenden Fehler bestimmt. Ein Fehler (exakter das Wirksamwerden eines Fehlers) ist definitionsgemäß das Ereignis, das im Verlust der Funktionsfähigkeit der entsprechenden Betrachtungseinheit liegt. Bei Rechnern können Fehler in der Hardware, in der Software und als Wechselwirkungsfehler (z. B. Bedienfehler) mit der Umwelt auftreten.

Die „klassischen" Fehler sind physikalischer Natur. Meist wird nach der Dauer zwischen permanenten und transienten Fehlern (Störungen) unterschieden, wobei letztere bei aus hochintegrierten Schaltkreisen aufgebauten Systemen dominieren. Die Tolerierung physikalischer Fehler – vor allem transienter Fehler – stellt für fehlertolerante Systeme eine „Standardforderung" dar.

Neben der Tolerierung physikalischer Fehler rückt zunehmend das sehr schwierige Problem der Tolerierung nichtphysikalischer Fehler (Entwurfsfehler der Hardware und vor allem der Software, Wechselwirkungsfehler) in den Mittelpunkt des Interesses.

Redundanz

Redundanz ist die Gesamtheit der Mittel, die ein fehlertolerantes System gegenüber einem fehlerintoleranten System enthalten muß, um seine Aufgaben erfüllen zu können. Man unterscheidet zwischen Hardware-, Software- und Zeitredundanz. Die Auswahl der zum Einsatz kommenden Redundanz ist abhängig von den zu tolerierenden Fehlern und weiteren technischen und ökonomischen Forderungen an das System.

Hardwareredundanz ist nicht nur eine meist erforderliche (z. B. zum Tolerieren von permanenten Hardwarefehlern), sondern oft auch eine effektive Redundanztechnik. Hardwareredundanz erfordert aber die Bereitstellung angepaßter Schaltkreise, da sonst meist keine praktikable Lösung möglich ist. Die zwei Formen der ***Hardwareredundanz*** sind:

Statische Redundanz

Das System enthält mehrere gleichartige und gleichzeitig aktive Komponenten, so daß durch einen Ergebnisvergleich nur in einzelnen Komponenten auftretende Fehler maskiert (d. h. unterdrückt) werden können (Maskenredundanz). Am bekanntesten ist das 2-aus-3-System, das Einzelfehler maskiert.

Dynamische Redundanz

Das System enthält mehrere gleichartige Komponenten, von denen die redundanten nur im Fehlerfall aktiv in Betrieb genommen werden.

Softwareredundanz ermöglicht das Nachrüsten von Fehlertoleranz in fehlerintolerante Systeme. Allerdings können dann nicht alle Fehler – beispielsweise keine permanenten Hardwarefehler – toleriert werden. Softwareredundanz ermöglicht die Realisierung komplexer Funktionen der Fehlererkennung, -lokalisierung und -korrektur. Allerdings sind dann die Laufzeiten der entsprechenden Programme wie auch eventuell in ihnen vorhandene Softwarefehler zu beachten.

Zeitredundanz besteht in der Wiederholung von Befehlen, Programmteilen oder ganzen Programmen und dem Vergleich der Ergebnisse. Damit können beispielsweise transiente Hardwarefehler toleriert werden. Zeitredundanz ist i. a. nur bei geringen Anforderungen einsetzbar.

Eine Sonderform der Redundanz ist die ***Koderedundanz***. Sie ermöglicht mittels fehlererkennender und fehlerkorrigierender Kodes die Korrektur von Fehlern, die bei der Übertragung oder Speicherung von Informationen entstehen. Das wohl bekannteste Beispiel ist die Einzelbitfehlerkorrektur bei dynamischen Speichern. Koderedundanz ist aber auch bei der Informationsverarbeitung einsetzbar.

Fehlererkennung

Die in fehlertoleranten Rechnersystemen implementierten Algorithmen zur Überwachung/Fehlererkennung hängen im starken Maße von den Forderungen an die Zuverlässigkeit ab.

Bei geringen Anforderungen kann man sich mit softwarerealisierten periodischen Selbsttest begnügen. Diese Selbsttests realisieren einen von innen nach außen fortschreitenden Test des Rechners. Dieses Verfahren ist in der Literatur ausführlich beschrieben worden bis hin zu Veröffentlichungen von geeigneten Testprogrammen und Methoden zu ihrer Erstellung /1/, /2/.

Der Nachteil dieses Verfahrens ist, daß zwischen den Tests keine Überwachung erfolgt, so daß es meist nur für Anfangstestung, Off-line-Testung sowie Testung nicht in Betrieb befindlicher Einheiten (Redundanztestung) geeignet ist. Für die On-line-Testung scheidet es i. a. aus.

Bei höheren Anforderungen an die Fehlererkennung ist u. U. ein erheblicher Hardwareaufwand erforderlich, den man in der Vergangenheit meist zu vermeiden suchte. Angesichts der Kostendegression für Hardware und der entgegengesetzten Tendenz für Software rücken Hardwaremaßnahmen aber zunehmend in den Mittelpunkt des Interesses.

Die Anwendung statischer Redundanz mit ihren fehlererkennenden und fehlerkorrigierenden Eigenschaften ist ein sehr leistungsfähiges Verfahren, dürfte jedoch als aufwendigste Lösung für die meisten Anwendungsfälle ausscheiden.

Die Duplizierung verbunden mit Parallelbe-



trieb und Ergebnisvergleich sowohl von Rechnern als auch nur von Prozessoren erscheint gegenwärtig bei komplexeren Strukturen das optimale Verfahren zur lückenlosen Erkennung von Hardwarefehlern zu sein. Einige Hersteller bieten bereits angepaßte Schaltkreise an. Ein Beispiel hierfür sind die INTEL-Prozessoren iAPX 432, die ein unmittelbares Zusammenschalten zweier Schaltkreise ermöglichen und mögliche Fehler nach außen signalisieren /3/, /4/.
Einen geringeren Aufwand als die Duplizierung verspricht die Anwendung fehlererkennender (meist arithmetischer) Kodes bei Prozessorschaltkreisen. Hierzu sind in letzter Zeit eine Reihe von Veröffentlichungen zu möglichen Kodes und Prozessorstrukturen erschienen /5/, /6/. Es ist aber nicht bekannt, ob derartige Schaltkreise bereits angeboten werden.
Neben den beschriebenen Verfahren werden zunehmend weitere Hardwareverfahren zur Überwachung einzelner Funktionen von Rechnern eingesetzt, wie beispielsweise Laufzeitüberwachung von Programmen, Überwachung von Speicherzugriffen und Berechtigungen usw. Diese Verfahren werden zunehmend in die Systemschaltkreise integriert. Mit ihrer Hilfe können indirekt einzelne Hardwarefehler erkannt werden, insbesondere sind sie jedoch zum Erkennen von Softwarefehlern geeignet.
So wurden z. B. in den INTEL-Prozessorschaltkreis 80386 ein Hardwareselbsttest (er erfaßt etwa 50 % der Struktur), 6 Debugregister zur Softwarefehlersuche und die Überwachung von Speicherzugriffen mittels interner MMU integriert /7/.

Diagnose

Nach dem Erkennen eines Fehlers kann durch Wiederholung des fehlerhaft verlaufenen Programmabschnitts bzw. durch Wiederholung des Tests, der zur Fehlererkennung geführt hat, entschieden werden, ob der betreffende Fehler transient oder permanent ist. Zur Lokalisierung von permanenten Fehlern muß ein Diagnoseprogramm gestartet werden. Dieses Programm ist im Prinzip gleich der Anfangstestung. Der wesentliche Unterschied besteht aber darin, daß die Zeit bei der Off-line-Testung sehr begrenzt ist. Das erfordert nicht zu lange Programmlaufzeiten des Diagnoseprogramms. In Mehrrechnersystemen sollte die Optimierung der Diagnose auf die Rechnerpausen oder die Zeit zwischen zwei Tasks erfolgen /8/.
Wird mehr Zeit für die Diagnose gefordert, muß der Rechner bzw. der Rechnermodul vom eigentlichen Rechenbetrieb entbunden werden. Damit ist der Rechner für diese Zeit nicht in der Lage, die anstehenden Aufgaben zu erfüllen.
Ziel der Diagnose ist es, daß Hardwareausfälle in den Systemkomponenten einer Rechnerkonfiguration erkannt und bis auf ersetzbare Einheiten herunter lokalisiert werden. Eine Meldung nach außen erleichtert das Auffinden der defekten Systemkomponenten wesentlich. Bereits bei nicht fehlertoleranten Systemen kann damit die Reparaturzeit verkürzt und die Verfügbarkeit erhöht werden. In einem fehlertoleranten System ist die Diagnose Voraussetzung für fehlertolerantes Verhalten. Hier wird das Diagnoseergebnis zur automatischen Rekonfiguration des Systems verwendet. In Mehrrechnersystemen wird zum Beispiel von einem dreistufigen Diagnoseansatz ausgegangen:

- Selbstdiagnose innerhalb eines Rechnermoduls
- Nachbarschaftsdiagnose zwischen unmittelbar miteinander verbundenen Modulen
- systemweite Diagnose.

Wird ein Rechnermodul ohne seine Verbindung zu anderen Modulen betrachtet, so stellt er einen typischen Mikrorechner mit CPU, Speicher und E/A-Schnittstelle dar. Die Diagnose von Hardwareausfällen kann durch fehlererkennende Kodes (spezielle Hardware ist erforderlich) oder durch Selbsttestprogramme erfolgen. Nach jedem Einschalten oder Rücksetzen eines Moduls laufen diese Selbsttestprogramme vollständig ab. Später werden sie während des Rechenbetriebes abgearbeitet. Wird Echtzeitverarbeitung gefordert, müssen die einzelnen Programmlaufzeiten beachtet werden (on-line Schranke).
Bei der Nachbarschaftsdiagnose greift ein Modul direkt auf einen anderen Modul zu und informiert sich über aufgetretene Fehler bzw. über das Ergebnis des Selbsttests. Auch das Verschicken von „I'm alive"-Nachrichten an benachbarte Module ist möglich. An dieser Stelle kann aber nicht entschieden werden, ob der Nachbarmodul defekt oder die Verbindung zu ihm unterbrochen ist. Zur Unterstützung der Rekonfiguration des Systems können die Nachbarn die Möglichkeit besitzen, einen als fehlerhaft erkannten Modul zurückzusetzen oder stillzulegen.
Aufgabe der systemweiten Diagnose ist es, einen als fehlerhaft erkannten Modul allen anderen Modulen mitzuteilen. An dieser Stelle kann auch die Entscheidung getroffen werden, ob ein Modul oder nur die Verbindung zu ihm defekt ist. Es liegen nun alle Informationen für eine Rekonfiguration des Systems vor /9/, /10/, /11/.

Rekonfiguration

Der Rekonfigurationsalgorithmus beinhaltet alle Aktionen, die nach dem Auftreten und Erkennen eines Fehlers durchgeführt werden mit dem Ziel, wieder zu einem normalen Rechenbetrieb zu gelangen. Automatische Rekonfigurationsalgorithmen lassen sich nach dem Status einteilen, den das System nach der Rekonfiguration (Systemwiederherstellung) besitzt.

Vollständige Rekonfiguration

Das System kehrt nach einer bestimmten Zeit mit seiner vollen Rechnerkapazität von der Fehlerbehandlung zurück.
Es ist ein Ersatz von defekten Hardwareelementen, und eine Korrektur der Daten erfolgt. Die Korrektur der Daten kann durch fehlerredundante Kodes, durch die Rückwärtsbewegung des Systems oder die Vorwärtsbewegung erfolgen. Bei dem Prinzip der Rückwärtsbewegung (Check-pointing) geht das System in einen fehlerfreien Zustand, der in der Vergangenheit liegt, zurück. Dazu werden im Programm Wiederaufsetzpunkte definiert und der Zustand des Systems abgespeichert. Im Fehlerfall wird ausgehend von dem zuletzt abgespeicherten Zustand ein Backup-Prozeß aktiviert. Dieser läuft in Mehrrechnersystemen auf einem anderen Rechnermodul und kann die Aufgaben des Programms lückenlos übernehmen.
Bei der Vorwärtsbewegung geht das System in einen fehlerfreien Zustand, der in der Zukunft liegt, wenn Fehler im System durch Rücksetzen der betroffenen Prozesse nicht maskiert werden können. Die Vorwärtsbewegung ist aber sehr schwer zu beherrschen.

Verringerte Rekonfiguration

Das System kehrt zu einem fehlerfreien Status zurück, aber mit verringerter Rechnerkapazität. Ursache hierfür kann sein, daß keine Reservemodule mehr vorhanden sind, daß Daten oder Programmteile verloren gingen und eine Reparatur länger als die zur Verfügung stehende Zeit dauert.
Es ist eine partielle Fehlertoleranz erfolgt.

Stillegung

Die Stillegung wird durchgeführt, wenn die verbleibende Rechnerkapazität unterhalb eines akzeptablen Minimums liegt. Sie hat das Ziel, eine Zerstörung von verbleibenden Daten und funktionstüchtigen Systemelementen zu verhindern und andere angeschlossene Systeme nicht zu stören. Ergebnisse, die bis zum Auftreten und Erkennen des Fehlers vorlagen, werden gerettet.
Die Systemwiederherstellung kann durch die Hard- oder Software übernommen werden. Im ersten Fall ist eine spezielle Hardware erforderlich. Der prinzipielle Vorteil dieser Methode liegt in der Unabhängigkeit von Operationen, die nach dem Auftreten des Fehlers durchgeführt werden.
Im zweiten Fall muß eine spezielle Software vorhanden sein. Der wesentliche Vorteil dieser Methode ist darin zu sehen, daß bei Vorhandensein einer zweiten identischen Hardware die Systemmodule genutzt werden können, um die Fehlertoleranz zu organisieren.
Bei der Anwendung der statischen Redundanz ist Softwareunterstützung notwendig, wenn das System nach dem Auftreten eines Fehlers weiterhin ordnungsgemäß arbeiten und dazu repariert werden soll /1/, /12/, /13/.

Literatur

/1/ Ebel, B.: Mikroprozessorselbsttest. Elektronische Rechenanlagen 20 (1978) 4, S. 186
/2/ Hunger, A.: Mikroprozessor-Selbsttest auf der Basis des Befehlssatzes. Elektronische Rechenanlagen 24 (1982) 1, S. 8
/3/ Johnson, D.: INTEL iAPX 432 – VLSI Building Blocks for a Fault-Tolerant Computer. Proc. NCC 1983, AFIPS Anaheim (1983) 5
/4/ Johnson, D.: The INTEL iAPX 432: A VLSI Architecture for Fault Tolerant Computer Systems. Computer (1984), S. 40
/5/ Geisselhardt, W.; Trautwein, M.: Fehlertolerante Mikroprozessorsysteme. Opladen 1984
/6/ Moritzen, K.: Entwurfsmethodik für Mikroprozessor-Selbstdiagnoseprogramme. Studienarbeit am IMMD III, Universität Erlangen 1981
/7/ Duzy, P.; Schallenberger, B.; Wallstab, S.; Moderne Mikroprozessoren. Elektronik München (1986) 14, S. 42
/8/ Avizienis, A.: Fault-Tolerance: The Survival Attribut of Digital Systems. Proceedings of the IEEE 66 (1978) 10, S. 1109
/9/ Moritzen, K.; Wirl, K.: Verteilte Diagnose auf dem Dirmu Multiprozessor-System. Workshop Fehlertolerante Mehrprozessor- und Mehrrechnersysteme, Erlangen 1984
/10/ Maehle, E.; Joseph, H.: Selbstdiagnose in Fehlertoleranten Dirmu-Multiprozessorkonfigurationen. Fehlertolerierende Rechnersysteme – GI-Fachtagung, Berlin-Heidelberg 1982
/11/ Bernhardt, D.; Klein, A.: Das fehlertolerierende Mehrrechnersystem BFS. Fehlertolerierende Rechnersysteme-GI-Fachtagung, Berlin-Heidelberg 1982
/12/ Herrman, F.: Das fehlertolerante Informationssystem 8832 / Das Fehlertoleranzkonzept. Software-Fehlertoleranz und -Zuverlässigkeit, Berlin-Heidelberg 1984
/13/ Schulz, A.: Das fehlertolerante System TANDEM T16. Software-Fehlertoleranz und -Zuverlässigkeit, Berlin-Heidelberg 1984

KONTAKT

Ingenieurhochschule Wismar, Sektion TdE/E, Dr. sc. Porep, Philipp-Müller-Straße, Wismar, 2400; Tel. 5 32 82

Fehlertolerante Mikrorechner – Funktionseinheiten mit der Redundanzart Graceful Degradation

Prof. Dr. Werner Kriesel
Technische Hochschule Leipzig, Sektion Automatisierunganlagen
Dr. Michael Schäfer
VEB Chemieanlagenbaukombinat Leipzig-Grimma, Stammbetrieb, HA CLG-Elektronik

Die Zuverlässigkeit von modernen Mikrorechner-Funktionseinheiten wird nicht allein durch die Zuverlässigkeit der zu ihrem Aufbau eingesetzten hochintegrierten mikroelektronischen Bauelemente bestimmt. Obwohl diese VLSI-Bauelemente über sehr hohe Einzelzuverlässigkeiten verfügen (z. B. Einchip-Mikrorechner U 8810: MTBF etwa 100 Jahre /1/), ist – bedingt durch die Komplexität von Automatisierungssystemen /2/ – die resultierende Zuverlässigkeit relativ gering (z. B. Grundgeräte von Reglersystemen: MTBF etwa 1 Jahr).
Erforderliche Maßnahmen zur Zuverlässigkeitserhöhung folgen entweder der **perfektionistischen Strategie** (mit dem Ziel, mögliche Fehlerursachen von vornherein zu vermeiden) oder der **Fehlertoleranz-Strategie,** welche mit Redundanz die nachteiligen Auswirkungen nicht vermeidbarer Fehler kompensiert. Zur Erzielung einer hohen Zuverlässigkeit müssen beide Strategien gleichzeitig angewendet werden.
Unter Beachtung ökonomischer Restriktionen für Automatisierungsmittel bestehen bei der Fehlertoleranz-Strategie noch Reserven hinsichtlich effektiver Redundanzarten, die für eine entscheidende Verbesserung der Zuverlässigkeit bei günstigem Preis/Leistungsverhältnis genutzt werden können. Nachfolgend wird auf diesen Aspekt näher eingegangen, wobei die Redundanzart Graceful Degradation im Mittelpunkt steht. Diese Redundanzart erfordert keine Mehrfachauslegung von Mikrorechner-Automatisierungsmitteln, sondern ist bei geringer Hardware-Erweiterung weitgehend softwareorientiert.

Fehlertoleranz

Eine Analyse des nationalen (vgl. /2/, /3/, /4/ u. /5/) sowie des internationalen /6/ Angebotes an Automatisierungsmitteln (ATM) zeigt sehr deutlich, daß fehlertolerante Systeme aus ökonomischen Erwägungen bisher nahezu nur bei erhöhten Sicherheitsanforderungen praktisch angewendet werden. Durch den stark zunehmenden Einsatz von ATM in komplexen CAD/CAM-Systemen stieg 1984 und 1985 die jährliche Produktionszuwachsrate um etwa 100 %, und bis 1990 werden jährliche Wachstumraten von 150 % erwartet /6/. Demgegenüber betragen die Produktionszuwachsraten seit 1983 bei der Produktionsautomatisierung insgesamt etwa 20–25 % pro Jahr /7/. Diese generell hohen Zuwachsraten bei Automatisierungsmitteln erfordern durch die Breitenanwendung zugleich neuartige Zuverlässigkeitskonzepte, u. a. Fehlertoleranz bei Mikroprozessor-Funktionseinheiten.
Voraussetzung für Fehlertoleranz ist, daß das System über gewisse Redundanz verfügt. In /8/ sind die wichtigsten Klassifikationsmerkmale für fehlertolerante Systeme dargestellt. Aus diesen Klassifikationsmerkmalen lassen sich die wichtigsten Redundanzarten ableiten:

- Majoritätssysteme (Spezialfall 2-von-3-System)
- Stand-by-Stand-Systeme
- definierte Leistungsminderung (Graceful Degradation).

Komplexe Automatisierungssysteme bestehen i. a. aus mehreren intelligenten Funktionseinheiten, wie z. B. bei /8/, /9/, /10/ sowie /11/ dargestellt. Dabei ist entsprechend den unterschiedlichen Anforderungen eine Kombination der unterschiedlichen Redundanzarten für die einzelnen Komponenten möglich und notwendig. In Bild 1 ist, ausgehend von Darlegungen in /6/, /9/, /10/, /11/ und /12/, eine mögliche Automatisierungssystem-Struktur speziell unter dem Gesichtspunkt von Fehlertoleranz dargestellt. Daraus sind die unterschiedlichen Redundanzarten in Funktionseinheiten mit zugeordnetem Feldbus ersichtlich. Weiterhin wird aus Bild 1 deutlich, daß der eigentliche Rechnerkern (Prozessor, Speicher, Prozeß-E/A-Scnnittstelle, Kommunikations-Schnittstelle) unabhängig vom Typ der Funktionseinheit mit der Redundanzart Graceful Degradation realisiert wird.
Eine vergleichende Bewertung der Redundanzarten nach quantitativ erzielbarem Zuverlässigkeitsgewinn wurde in /12/ vorgenommen. Daraus geht hervor, daß die Redundanzart Graceful Degradation überall dort eine große Bedeutung erlangen wird, wo im Fehlerfall eine Leistungsminderung vertretbar und für die erforderliche Zuverlässigkeitserhöhung nur geringe Mehraufwendungen zulässig und möglich sind.
Damit muß Fehlertoleranz nicht auf ausgewählte Einsatzfälle begrenzt bleiben, sondern kann für zukünftige ATM zum selbstverständlichen Qualitätsmerkmal werden. Die Redundanzart Graceful Degradation wird daher im Folgenden näher betrachtet.

Redundanzart Graceful Degradation

Entsprechend den in /6/ genannten Klassifikationsmerkmalen ist die Redundanzart Graceful Degradation durch aktive Funktionsbeteiligung und dynamische Fehlerkompensationsverfahren charakterisiert, wobei je nach Struktur des Automatisierungsmittels entweder eine zentrale oder dezentrale lokale funktionelle Konzentration vorliegt.
Der prinzipielle Aufbau des redundanten Systems ist in Bild 2 dargestellt. Um ein System fehlertolerant zu realisieren, sind gegenüber einem nicht-fehlertoleranten System zusätzlich die Funktionen

- Fehlerdiagnose
- Rekonfiguration
- Wiederanlauf (recovery)

zu implementieren. Dies kann bei der Redundanzart Graceful Degradation mit relativ geringem Hardware-Mehraufwand erfolgen, weil hier Fehlertoleranz vorwiegend mit Softwaremitteln realisiert wird. Diese Mehraufwendungen können hauptsächlich softwareseitig realisiert und mehrfach genutzt werden, so daß ein günstiges Aufwand-Nutzen-Verhältnis entsteht. Besonders die Komponenten für Fehlerdiagnose, Wartung und Service eignen sich gut für multivalente, applikationsunabhängige Nutzung.
Unter der in Bild 2 dargestellten Modulmenge sind sowohl Hardwarebaugruppen als auch Softwarekomponenten zu verstehen. Aus Bild 2 wird ersichtlich, daß die Leistungsminderung im Fehlerfall durch Suspendierung der defekten Module entsprechend der Rekonfigurationsstrategie erfolgt.
Je nach Art und Menge der aufgetretenen Fehler kann das System verschiedene Zustände geringerer Leistungsfähigkeit einnehmen. Die Leistungsminderung kann sowohl durch verringerten Umfang von Hardwarebaugruppen (z. B. E/A-Leitungen, Speichergröße) als auch durch Suspendierung bestimmter Softwarefunktionen (z. B. Optimierung) erreicht werden.
Wie aus Bild 2 ersichtlich, wird Fehlertoleranz in sequentiellen Arbeitsschritten erreicht:

- Fehlerdiagnose
- Rekonfiguration
- Wiederanlauf.

Der Fehlerdiagnose kommt in diesem Zusammenhang die größte Bedeutung zu, weil nur auf solche Fehlerursachen reagiert werden kann, die zuvor diagnostiziert wurden. Weiterhin erfordert Fehlerdiagnose den größten Aufwand innerhalb der Fehlertoleranzmaßnahmen. Auf Fehlerdiagnose wird daher im folgenden Abschnitt noch näher eingegangen.
Rekonfiguration ist die Fähigkeit des fehlertoleranten Systems, aufgrund des Diagnose-Ergebnisses seine Struktur zu ändern. Wiederanlauf ist die Fortsetzung der Arbeit des Systems nach der Strukturänderung unter definierten Bedingungen und mit gültigen Informationen.

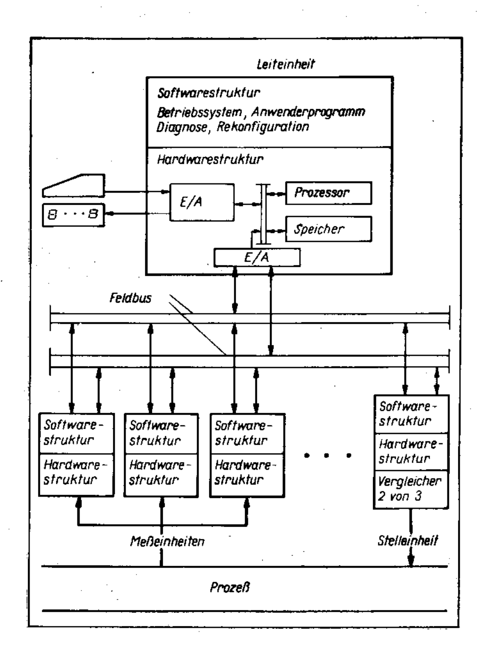


Bild 1 Redundanzstruktur eines Automatisierungssystems

Fehlerdiagnose

Fehlerdiagnose muß bei Echtzeitsystemen in zwei Phasen – Fehlererkennung und Fehlerlokalisierung – unterteilt werden. Während die Fehlererkennung im Hintergrund zur eigentlichen Automatisierungsfunktion abläuft (On-line-Diagnose), wird während der Fehlerlokalisierung das Automatisierungsobjekt durch Automatisierungsmittel auf einem definierten Zustand gehalten; die Automatisierungsfunktion wird im Hintergrund realisiert. Diese Phase wird als Off-line-Diagnose bezeichnet.

Fehlererkennung

Die Leistungsfähigkeit der Fehlererkennung ist primär an der Fehlererkennungswahrscheinlichkeit und der Ausfallerkennungszeit meßbar. Bei der Zielstellung für die Fehlererkennungswahrscheinlichkeit ist festzulegen, welche Fehlerursachen qualitativ und quantitativ diagnostiziert werden sollen. Für die Festlegung der benötigten Testmittel sind die diagnostizierbaren Fehlerursachen nach dem Fehlerort zu klassifizieren, weil völlig unterschiedliche Verfahren notwendig sind:

- rechnerinterner Fehlerort
(Bus, Speicher, Mikroprozessor, E/A-Schnittstelle, Software)
- Meß- und Stelleinheiten
- externe Speicher
- Automatisierungsobjekt.

Die Ausfallerkennungszeit wird durch folgendes bestimmt: Die verfügbare Zeit für On-line-Diagnose ist wegen der Sicherung des Echtzeitverhaltens, speziell bei Einprozessor-Systemen, begrenzt. Das führt zu einer zeitgeschachtelten Diagnose (wie sie z. B. bei /13/ beschrieben ist), wobei die maximale Ausfallerkennungszeit nicht überschritten werden darf. Die maximale Ausfallerkennungszeit ist primär von den Anforderungen des Prozesses bestimmt und sekundär von der Zuverlässigkeit des Automatisierungsmittels. Speziell bei Anlagen der chemischen Industrie können z. T. sehr harte Forderungen wegen hoher ökonomischer Schäden bei Verlust der Produktqualität entstehen.
Ist die Forderung durch die gewählte ATM-Struktur nicht realisierbar, so ist deren Struktur zu verändern (von Ein-Prozessor- zu Mehrprozessorsystemen bzw. schnellen 16-Bit-Prozessoren). So übernimmt z. B. bei Mehrrechnersystemen ein Prozessor vorwiegend Diagnose-Aufgaben, wie dies bei /10/ dargestellt ist.

Fehlerlokalisierung

Entsprechend der vorgesehenen Reparaturstrategie besteht das Ziel darin, nur so detailliert wie nötig nicht das einzelne Element, sondern den fehlerverursachenden Modul zu lokalisieren. Die Modulgröße wird von der Größe der austauschbaren Baugruppen bestimmt. Wegen des erforderlichen Zeitaufwandes ist Fehlerlokalisierung i. a. nicht als On-line-Diagnose ausführbar.
Nach Lokalisierung einer Fehlerursache ist eine Bewertung hinsichtlich der Tolerierbarkeit erforderlich. Bei nichttolerierbaren Fehlern wird mittels der Rekonfigurationseinrichtung ein Fehlerzustand mit definiertem Verhalten des Automatisierungsobjektes realisiert. Bei tolerierbaren Fehlern ist für die Wahl der zweckmäßigen Rekonfigurationsstrategie entscheidend, ob permanente oder transiente Fehlerursachen vorliegen. Führt Fehlerlokalisierung zu keinem Ergebnis, ob-

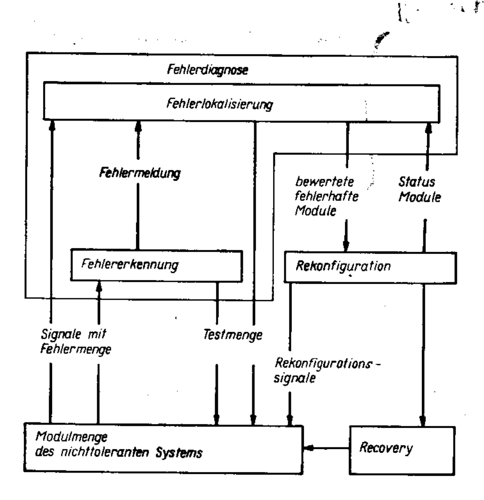


Bild 2 Prinzipieller Aufbau fehlertoleranter Systeme

wohl bei On-line-Diagnose ein Fehler erkannt wurde, wird auf eine transiente Fehlerursache geschlossen.
Jeder diagnostizierte Fehler führt zu einer Fehlermeldung an das übergeordnete System, wobei folgende Informationen erforderlich sind:

– ***Fehlerursache***
ausgefallener Modul
Ausfallzeitpunkt (soweit möglich)
– ***Bewertungsergebnis***
Tolerierbarkeit
Zeitdauer (permanent/transient)
– ***Rekonfigurationsergebnis***
suspendierte bzw. leistungsgeminderte Funktion
– ***Zustand der Funktionseinheit*** (soweit erforderlich)
aktive/inaktive Module
– ***Fehlerstatistik*** (soweit erforderlich)
Zahl der Fehler je Modul.

Testverfahren

Die Qualität eines Testverfahrens ist meßbar an der Relation von erkannten zu tatsächlich aufgetretenen Fehlerursachen. Die in Betracht kommenden Testverfahren sind wie folgt klassifizierbar:

- strukturorientiert
- funktionsorientiert.

Nach /14/ sind mit strukturorientierten Tests Fehlererfassungszahlen bis zu 100 % und bei funktionsorientierten Tests bis etwa 75–90 % erzielbar. Für strukturorientierte Tests werden die Testdaten auf der Basis von Modellen mit niedrigem Abstraktionsgrad bereitgestellt. (z. B. Fehlermodell Gatterschaltung). Daher sind diese Tests vorwiegend für Hardwaremodule geringer Komplexität geeignet. Dagegen werden für funktionsorientierte Tests die Testdaten auf der Basis abstrakter Modelle (z. B. Registertransfer bzw. Automatenmodell) bereitgestellt. Diese Verfahren werden daher bevorzugt für komplexe Module (Software, Mikroprozessoren) angewandt. Eine Bewertung nach Aufwand und Ergebnis ist in Tafel 1 gegeben. Bezüglich weiterführender Angaben wird auf /15/, /16/ und /17/ verwiesen.

Fehlerdiagnose für intelligente Funktionseinheiten

Die nachfolgenden Ausführungen basieren auf Ergebnissen bei der Entwicklung einer intelligenten Meß- und Steuereinrichtung für Tankstellen mit hohen Zuverlässigkeits-, jedoch ohne besondere Sicherheitsanforderungen. Kernstück ist ein Automatisierungsmittel mit einem einzigen Einchipmikrorechner UB 8820, 6 KByte ROM, 1 KByte RAM, fünf Funktionstasten, 19stelliger LCD-Anzeige, fünf digitalen und drei analogen Eingängen, drei digitalen Ausgängen und einer seriellen Schnittstelle zur zentralen Überwachung, Bilanzierung und Abrechnung an eine übergeordnete Leiteinrichtung.
Untersuchungen der Zuverlässigkeit nach /12/ sowie die Felddaten der Praxiserprobung zeigen sehr deutlich, daß die Fehlerdiagnose sowohl rechnerinterne Fehler, Fehler an Meß- und Stelleinheiten als auch Fehler des Automatisierungsobjektes diagnostizieren muß. Softwarefehler konnten vernachlässigt werden, da diese aufgrund sehr aufwendiger Erprobungen im Rahmen der erforderlichen Zulassungsverfahren beseitigt wurden.
In Tafel 2 ist eine Zusammenfassung der diagnostizierbaren Fehlerursachen mit Bewertung des realisierten Testverfahrens und Angabe des erzielten Fehleraufdeckungsgrades dargestellt. Bei tolerierbaren Fehlern ist eine Weiterarbeit der Meßeinrichtung bei gezielter Leistungsminderung möglich. Auf einen Selbsttest des Prozessors (bei komplexen Systemen empfehlenswert) mußte aus Aufwandsgründen verzichtet werden. Bei nicht-tolerierbaren Fehlerursachen reagiert das System mit Übergang in einen Fehlerzustand (Automatisierungsobjekt im sicheren Zustand), wobei eine Mitteilung über die Fehlerursache an die übergeordnete Leiteinrichtung gesendet wird. In die intelligente Funktionseinheit ist noch eine zusätzliche Diagnose integriert, welche die Inbetriebnahme und Wartung des Automatisierungssystems ohne externes Servicegerät ermöglicht.

Ausblick

Automatisierungsmittel auf Mikrorechner-Basis müssen in Zukunft generell in weit höherem Maße als bisher auch mit Eigenfunktionen für Zuverlässigkeitsaufgaben ausgestattet werden. Hierfür sollte der volle Umfang ökonomisch tragfähiger Redundanzformen ausgeschöpft werden.
Um den Zusatzaufwand für Redundanzmaßnahmen in vernünftiger Relation zum Grundaufwand der jeweiligen Funktionseinheit zu halten, macht sich ein aufwandsgestuftes und damit leistungsgestuftes Redundanzkonzept mit mehreren Ebenen erforderlich. Es sollte von den kleinsten intelligenten Bausteinen über die Geräteebene bis zur Anlagenebene reichen.
Mit dem angeführten praktischen Beispiel konnte nachgewiesen werden, daß selbst für Funktionseinheiten mit nur einem Einchipmikrorechner durch Anwendung der Redundanzart Graceful Degradation ohne nennenswerte Hardwareerweiterung ein spürbarer Zuverlässigkeitsgewinn durch Fehlertoleranz erzielbar ist. Hieraus entstehen zugleich auch Vorteile für die Prüfung bei der Fertigung, der Inbetriebnahme sowie der Instandhaltung.
Künftig setzt sich dieser skizzierte Trend bis in die Ebene mikroelektronischer Bauelemente fort. Es sind bereits VLSI-Schaltkreise bekannt geworden, in die Fehlertoleranz mit integriert ist.

✉ KONTAKT ☏

VEB Chemieanlagenbaukombinat Leipzig-Grimma, Bereich CLG-Elektronik, Dr. M. Schäfer, Tel. Grimma 63/36 82

Tafel 1 Bewertung der Fehlerstrategien

Merkmal	Teststrategie strukturorientiert Ergebnis (Aufwand)	funktionsorientiert Ergebnis (Aufwand)
Beschreibung des Prüflings	exakte Beschreibung (großer Aufwand)	starke Vereinfachung (mittlerer Aufwand)
Testmustergenerierung	(großer Aufwand)	(sehr großer Aufwand)
Testverifizierung	vollständige Fehlererfassung (kein Aufwand)	(sehr großer Aufwand)
Testausführung	lange Laufzeiten	kurze Laufzeiten
Fehlerlokalisierung	exakte Lokalisierung Bauelemente-Niveau	Modul-Niveau
Lokalisierbare Fehlerarten	Hardwarefehler	Hardware- und Entwurfsfehler

Tafel 2 Zusammenhang von Fehlerursachen und Testverfahren

Fehlerursache	Bewertung	Fehlermodell	Testverfahren	Fehleraufdeckungsgrad %
rechnerinterner Fehlerort				
Einchipmikrorechner	nicht tolerierbar	Registertransfer	Programmlaufkontrolle	20
ROM	(1)	Registertransfer	Prüfsumme	100
RAM	(1)	Registertransfer	musterabhängige Kopplung, Prüfsumme	100
Stromversorgung	nicht tolerierbar	–	Spannungskontrolle	50
serielle Datenübertrg.	tolerierbar	Registertransfer	Echobetrieb	100
E/A-Schnittstelle	(1)	stuck-at-0/1	Plausibilitätskontrollen	100
Meß- und Stelleinheiten				
Meßeinheiten	(1)	stuck-at-0/1	indirekt über Meßeinheiten	100
Stelleinheiten	(1)	stuck-at-0/1	indirekt über Meßeinheiten	20

Legende: (1) in Abhängigkeit von Anzahl der Fehlerursachen bzw. genauer Lage tolerierbar bzw. nicht tolerierbar

Literatur

/1/ Intergrierte Halbleiterschaltkreise U8810/U8811 Technische Bedingungen. TGL 37360 1982

/2/ EAW-Electronic S2000. Kombinat Elektro-Apparatewerke Berlin-Treptow, Zentrum für Forschung und Technologie, Berlin 1987

/3/ Häßner, H.: Projektierung der Zuverlässigkeit im System audatec. messen, steuern, regeln 30 (1987) 4, S. 151

/4/ Häßner, H.; Neumann, P.: Dimensionierung des Zuverlässigkeitsverhaltens von Mikrorechner-Automatisierungsanlagen. messen, steuern, regeln 30 (1987) 4, S. 176

/5/ Funktionsbeschreibung IRS711. VEB Numerik „Karl Marx" Karl-Marx-Stadt 1986

/6/ Kieser, H.; Bankel, M.: Einchipmikrorechner VEB Verlag Technik, Berlin 1986

/7/ Töpfer, H.; Fuchs, H.: Auswertung „Interkama '86". messen, steuern, regeln 30 (1987) 4, S. 186

/8/ Töpfer, H.; Kriesel, W. (Hrsg.): Funktionseinheiten der Automatisierungstechnik elektrisch, pneumatisch, hydraulisch. VEB Verlag Technik, Berlin 1987

/9/ Töpfer, H.; Kriesel, W.: Automatisierungstechnik – Gegenwart und Zukunft. Reihe Automatisierungstechnik, Bd. 200, VEB Verlag Technik, Berlin 1982

/10/ Kriesel, W.; Blum, H.; Telschow, D.: Intelligente und buskoppelbare Stelleinrichtungen. messen, steuern, regeln 28 (1985) 12, S. 534

/11/ Kriesel, W.; Richter, W.; Gibas, P.: Feldbusnetz für Automatisierungssysteme mit intelligenten Funktionseinheiten. messen, steuern, regeln 30 (1987) 11, S. 486

/12/ Schäfer, M.: Automatisierungsstrukturen mit Einchipmikrorechnern, Zuverlässigkeitsaspekt. Diss. (A), TH Leipzig 1987

/13/ Kriesel, W. Kirste, R.; Steinbock, K.: Partielle Fehlertoleranz für Mikrorechner-Funktionseinheiten. Mikroprozessortechnik 2 (1988) 6, S. 168

/14/ Nilsson, S.: Konzepte und Architektur eines fehlertoleranten Mehrmikrorechner-Systems. Diss., Universität Karlsruhe 1981

/15/ Reinert, D.: Entwurf und Diagnose komplexer digitaler Systeme. VEB Verlag Technik, Berlin 1983

/16/ Hunger, A.: Neues Testverfahren zum Selbststart von Mikroprozessoren. Diss., TH Aachen 1982

/17/ Hedtke, R.: Mikroprozessorsysteme – Zuverlässigkeit, Testverfahren, Fehlertoleranz. Springer Verlag, Berlin-Heidelberg-New York-Tokyo 1984

K-6313-Treiber für den KC 85/3 in der REM-Zeile

Die hier vorgestellte Lösung ist auf die wichtigsten Funktionen verkürzt. Sie hat gegenüber anderen Methoden einige Vorteile und genügt den herkömmlichen Ansprüchen der BASIC-Arbeit.

Die Startadresse wurde auf 0406H gelegt. Das Programm steht in drei zusammengelegten REM-Zeilen. Als Zeile „0" kann es mit jedem beliebigen BASIC-Programm „NAME" durch CLOAD „V24REM" und anschließendes CLOAD „NAME" gekoppelt werden, das lästige Nachladen der Treibersoftware entfällt. Ein weiterer Vorteil ist die gezielte Anwendung der Hardcopy-Funktion, die durch einen CALL-Befehl im Wechsel geschaltet werden kann (Analog zur Tastenkombination SHIFT + CLR).

Das Assemblerprotokoll (Bild 1) zeigt darüber hinaus einige Unterprogramme, die vom Nutzer in eigene Routinen einbezogen werden können.

Geeignet dazu ist die Befehlsfolge ab 04A6H ‚UPSTART'. Nach dem Zuschalten des IRM wird zu einer Adresse gesprungen, die ab 02FEH abgelegt wurde. Das so eingestellte Unterprogramm kann dann wiederholt mit CALL*4A6 aufgerufen werden (Handhabung wie bei USR(X)-Funktion).

Eine Testanwendung ist in Bild 2 dargestellt. Es zeigt sich, daß Test 3 und 4 zu gleichen Ergebnissen führen, PRINT#2 also oftmals entfallen kann.

Uwe Zierott

```
Copyright U. Zierott 1987

10 DOKE 766,1124        : ! STADR,HCOPY
20 CALL*406             : ! V24AN
30 PRINT#2"Test 1"
40 PRINT  "Test 2"
50 CALL*4A6             : ! HCOPY ein
60 PRINT  "Test 3"
70 PRINT#2"Test 4"
80 CALL*4A6             : ! HCOPY aus
90 PRINT  "ENDE !"
OK
```

2

```
;*************************************
;*  V24- Treiber (K6313) mit soft-  *
;*  waregesteuerter Protokollfunkt.*
;*    in einer BASIC- REM-Zeile     *
;*   (C) Uwe Zierott  / 18.11.87    *
;*************************************
         ORG      0406H
STADR    EQU      02FEH
OUTR     EQU      0B811H
HCOPY    EQU      0B799H
OUTAB    EQU      0B7B9H
UOUT     EQU      0B7BEH
PV1      EQU      0F003H
PV3      EQU      0F009H
;
;M003 suchen und initialisieren
V24AN    CALL     IRMAN
         LD       HL,(OUTAB)
         LD       (OUTR),HL
         LD       BC,00880H
MODSU    IN       A,(C)
         CP       0EEH
         JR       Z,MODDA
         CP       0FFH
         JR       NZ,WEITER
ERROR    CALL     PV1
         DEFB     19H
         RET
;
WEITER   LD       A,4
         ADD      B
         LD       B,A
         JR       MODSU
;
MODDA    LD       L,B
         LD       A,2
         LD       D,1
         LD       E,D
         CALL     PV1
         DEFB     26H
         CALL     ZEIGER1
         LD       HL,ZEIGER3
         LD       (HCOPY),HL
         LD       C,00CH
         LD       B,002H
         LD       HL,DRUCKI
         DI
         OTIR
         LD       BC,0060AH
         OTIR
         EI
         LD       A,0AH
         CALL     OUT2
         LD       A,0DH
;
;UP zur Ausgabe auf Anwenderkanal 1
OUT2     CALL     PV1
         DEFB     2
         RET
;
;Systemzeiger setzen
ZEIGER1 LD        HL,(OUTR)
        LD        DE,DRUCK1
ZEIGER2 LD        (OUTAB),HL
        LD        (UOUT),DE
        RET
;
;Protokoll ein/aus
ZEIGER3 PUSH      HL
        PUSH      DE
        LD        HL,(OUTAB)
        LD        L,(HL)
        INC       L
        DEC       L
        JR        NZ,ZEIGER5
        LD        DE,VIDEO
        LD        HL,MODDA+2
        CALL      ZEIGER2
ZEIGER4 POP       DE
        POP       HL
        RET
;
ZEIGER5 CALL      ZEIGER1
        JR        ZEIGER4
;
;Ausgabe eines Zeichens
VIDEO    CALL     CRT
DRUCK1   PUSH     AF
         PUSH     BC
DRUCK2   IN       A,00AH
         BIT      2,A
         JR       NZ,DRUCK3
         LD       A,001H
         CALL     PV1
         DEFB     014H
         JR       DRUCK2
DRUCK3   POP      BC
         POP      AF
         OUT      008H,A
         RET
;
;Zeichenausgabe auf Bildschirm
CRT      PUSH     AF
         XOR      A
         LD       E,A
         POP      AF
         CALL     PV3
         RET
;
;Einschalten IRM
IRMAN    IN       A,088H
         SET      2,A
         OUT      088H,A
         RET
;
;Aufruf eines UP mit Einschalten
;des IRM. Die Startadresse muß in
;02FEH stehen. (DOKE 766,STADR)
UPSTART CALL      IRMAN
        LD        HL,(STADR)
        JP        (HL)
;
DRUCKI   DEFW     05B47H
         DEFW     00404H
         DEFW     02003H
         DEFW     06A05H
```

Partielle Fehlertoleranz für Mikrorechner-Funktionseinheiten

Rosi Kirste
VEB Chemieanlagenbaukombinat Leipzig-Grimma, Stammbetrieb, CLG-Elektronik
Prof. Dr. Werner Kriesel,
Dr. Klaus Steinbock,
Technische Hochschule Leipzig, Sektion Automatisierungsanlagen

Problemstellung

In der Klein- und Mittelserien-Produktion werden in zunehmendem Umfang durchgängig automatisierte flexible Fertigungslinien benötigt /1/, die häufig unter Integration bereits vorhandener mikrorechnergesteuerter Arbeitsstationen aufgebaut werden. Für ausreichende technologische Verfügbarkeit derartiger Systeme und bedienarme Fahrweise ist Fehlertoleranz unverzichtbar /2/. Fehlertoleranz erfordert Redundanz. Fehlertoleranz auf Systemebene bedingt eine zuverlässige Kommunikation der Systemkomponenten /3/. Speziell in die Anlagenautomatisierung eingeführte Redundanzarten /4/ können infolge ökonomischer Restriktionen nicht übernommen werden.
Vorhandene mikrorechnergesteuerte Funktionseinheiten sind i. a. nicht fehlertolerant ausgelegt. Realisierungsformen für Fehlertoleranz sind z. B. bei /5/ angegeben. Bei /6/ wird vorgeschlagen, Funktionseinheiten nachträglich mit Fehlertoleranzeigenschaften auszustatten und dabei Hard- und Softwaremöglichkeiten gleichermaßen zu nutzen. Fehlertoleranz wird dabei schrittweise, in Realisierungsphasen (Diagnose, Rekonfiguration und Wiederanlauf) erreicht. In Weiterführung dieses Grundgedankens wird nachfolgend die Möglichkeit einer partiellen, modulbezogenen Realisierung von Fehlertoleranz vorgestellt.
Wesentliches Kriterium beider Methoden ist, daß mit vertretbarem Aufwand für das vorhandene System nachträglich Fehlertoleranz erreicht werden kann, ohne wesentliche Eingriffe in Hardware und Betriebssystem der vorhandenen Lösung vornehmen zu müssen.

Lösungsweg

Der Nutzen integrierter Fehlerdiagnose im Zusammenhang mit der für derartige Anwendungen besonders geeigneten Redundanzart *definierte Leistungsminderung* (Graceful Degradation) ist bei /6/ und /7/ dargestellt. Integrierte Fehlerdiagnose ist nachrüstbar; Beispiele hierzu sowie methodische Hinweise sind bei /8/ angegeben.
Fehlertoleranz eines Systems besteht aus einer Summe von Maßnahmen, die auf die Tolerierung jeweils ganz bestimmter Fehlerarten gerichtet sind, die ihrerseits nur durch ganz bestimmte Arten von Modulen verursacht werden können. Zwischen den Fehlern unterschiedlicher Modularten bestehen i. a. keine Zusammenhänge. Dies wird bei Untergliederung eines Systems nach Diagnose-Objekten deutlich. Es ist möglich, auch Rekonfiguration partiell zu realisieren.

Bei Neuentwicklungen ist die Fehlertoleranz-Strategie Bestandteil des Betriebssystems. Für die hier zu diskutierende Rekonstruktion vorhandener Systeme soll auf Eingriffe in Hardware und Betriebssystem weitgehend verzichtet werden. Hardwareänderungen werden weitgehend vermieden, indem Rekonfiguration ebenso wie Fehlerdiagnose softwareseitig realisiert wird (Anwendung software-implementierter Fehlertoleranz, vgl. /9/). Eingriffe in das vorhandene Betriebssystem lassen sich reduzieren, indem die Nachrüstung von Fehlertoleranzeigenschaften als Erweiterung mit überschaubaren, gut zu kontrollierenden Schnittstellen erfolgt. Das Wesentliche einer dazu geeigneten Vorgehensweise wird am Beispiel der Rekonfiguration deutlich.
Durch Rekonfiguration wird Redundanz nutzbar gemacht. Rekonfiguration bedeutet: Veränderung der Systemstruktur; es ändern sich Relationen zwischen vorhandenen Systemkomponenten. In ausführbaren Maschinenanweisungen ist die Systemstruktur auf Adressen abgebildet; Rekonfiguration erfordert Veränderung von Adressen (d. h., das System arbeitet anschließend unter Benutzung anderer Adressen weiter). Dazu müssen die betroffenen Adressen/Relationen identifizierbar sein. Bei vorhandenen Systemen erweist sich als Haupthindernis die i. a. völlige Regellosigkeit bei Zugriffen auf Informationen, Systemressourcen (bzw. Teilen davon, vgl. Bild 1). Die Lösung des Rekonfigurationsproblems erfordert zunächst eine Entflechtung und Ordnung der betroffenen Relationen (vgl. Bild 2).
Speziell auf derartige Problemstellungen wird in der Informatik das Prinzip der objektorientierten Vorgehensweise angewandt (vgl. /10/). Die Wirkung beruht darauf, daß ein *Objekt* mit allen in ihm möglichen Operationen zu einem Teilsystem zusammengefaßt wird. Seine innere Organisation ist gegenüber dem restlichen System hinter einer Schnittstelle verborgen. Wesentlich für das hier interessierende Rekonfigurationsproblem ist dabei, daß Strukturveränderungen des Teilsystems dem übrigen System verborgen bleiben und dort keine Auswirkungen haben. Um die systemtheoretische Grundlage dieses Prinzips nutzen zu können, ist folgendes zu beachten:
Objektklassen umfassen in der Informatik ausschließlich Daten (abstrakte Datentypen, Datenkapseln). Dies ist für die Lösung von Zuverlässigkeitsproblemen nicht ausreichend, da hier auch die Hardware betrachtet werden muß. In diesem Zusammenhang erscheint es zulässig, den Objektbegriff auf stoffliche bzw. energetische Träger von Informationen auszudehnen. Die Softwarekomponenten für Diagnose des Objektes und seine Rekonfiguration sind durch den Definitionsbereich der im Objekt zulässigen Operationen berücksichtigt.
Mit diesem Ansatz eignet sich die objektorientierte Vorgehensweise zur Lösung von Rekonfigurationsproblemen für solche Systemkomponenten, die sich sowohl hinsichtlich der von ihnen repräsentierten Daten als auch deren Träger mit Softwaremitteln (d. h. funktionell) abgrenzen lassen. Dies trifft z. B. zu auf das Hineinbringen von Redundanz in Datenspeicher oder von redundanter (diversitärer) Software.
Eine zweckmäßige Vorgehensweise zur Realisierung partieller Fehlertoleranz wird nachfolgend an einem Beispiel erläutert /8/.

Realisierungsbeispiel

Aus Gründen der Anschaulichkeit wurde als Beispiel ein *in sich fehlertoleranter* Speicher gewählt. Eine vorhandene mikrorechnergesteuerte Funktionseinheit soll nachträglich fehlertolerant gegenüber Speicherfehlern werden, um veränderte technologische Anforderungen zu erfüllen, die eine sichere Verfügbarkeit von Prozeßvariablen bedingen.
Der RAM des Systems wird – unabhängig von der Schaltkreis-Anordnung – als zusammengehöriger Speicherbereich aufgefaßt, er bildet mit den gespeicherten Daten das *Objekt*. Vorkommende *Operationen auf dem Objekt* sind
- Speichern und Lesen von Daten
- Diagnose auf Speicherfehler
- Rekonfiguration bei tolerierbaren Fehlern.

Alle dazu erforderlichen Softwarekomponenten müssen in einem Modul zusammengefaßt werden, dessen Funktionen aus dem übrigen System nur durch Parameterübergabe an einer Schnittstelle erreichbar sind.

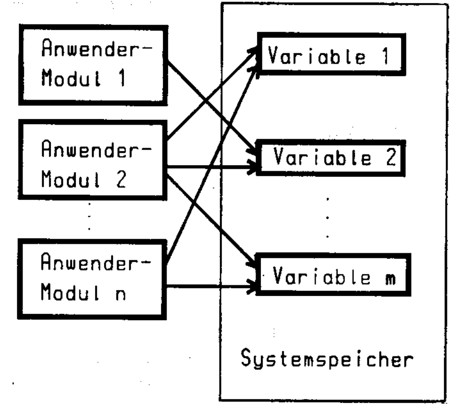


Bild 1 Ungeordneter Zugriff auf Speichervariable

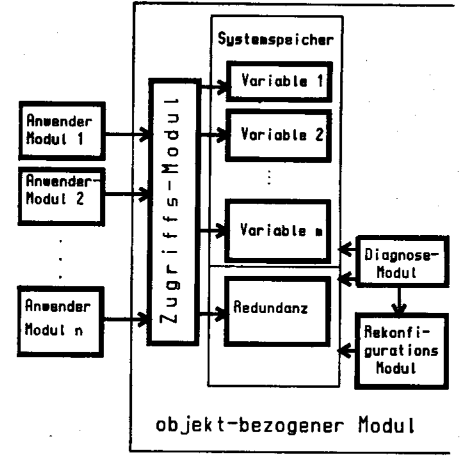


Bild 2 Entflechtung der Speicherzugriffe bei objektbezogener Vorgehensweise für einen in sich fehlertoleranten Systemspeicher /8/.

Diagnose und Rekonfiguration benötigen keine Parameter; sie geben an das vorhandene Betriebssystem Zustandsinformationen in der für Systemnachrichten vorgesehenen Weise. Ihre Nachrüstung bereitet bei ausreichenden Systemreserven keine Schwierigkeiten.
Lese- und Schreibzugriffe sind über das gesamte vorhandene System verteilt und erfolgen bei Assemblerprogrammierung i. a. völlige beliebig (vgl. Bild 1). Ihre nachträgliche Entkopplung erfordert, daß jeder *private* Zugriff im vorhandenen System in einen *Aufruf der Schnittstelle* des fehlertoleranten Speichers mit Übergabe von Parametern umgewandelt wird (vgl. Bild 2). Dabei kann die Registervorbelegung für den *privaten Speicherzugriff* i. a. als Parameter beibehalten werden, weil z. B. durch Anwendung von Testhilfen oder Speicherzugriffsroutinen häufig bereits ein Normierungseffekt gegeben ist.
Bei dem nach /8/ realisierten Beispiel wurde ein zerstörungsfreier GALPAT II-Test vorgesehen, der während des Echtzeitbetriebes zyklisch abläuft. Im Fehlerfall wird der Inhalt der fehlerhaften Speicherzelle in einen Redundanzbereich ausgelagert. Bei Zugriff auf die betroffene Variable wird automatisch der ausgelagerte Wert mit hineingebracht und der anfordernde Modul über eine mögliche Verfälschung informiert; er kann eine (problemabhängige) Plausibilitätsprüfung veranlassen.
Vom Auftreten des ersten Speicherfehlers an entsteht eine Verlängerung der Zugriffszeit in Abhängigkeit von der Anzahl der Fehler (Redundanzart: definierte Leistungsminderung). Bei den Angaben in Tafel 1 ist zu beachten, daß sie stark applikationsabhängig sind.
Wenn der Prozessor der Funktionseinheit für Speichertransfer einen speziellen Operationskode aufweist, wie im Beispiel der Einchipmikrorechner U 882, so kann der Austausch der Kodefolgen in den Quelltexten in einfacher Weise mit Unterstützung durch Entwicklungssystem-Komponenten erfolgen.
Wie das Beispiel zeigt, werden alle in dem neuen Modul zusammengefaßten Funktionen gleichermaßen zur Lösung herangezogen. Nachteilig bei dem gewählten Lösungsprinzip ist eine Verlängerung der Antwortzeit durch Einführung einer zusätzlichen inneren Systemgrenze. Bei dem fehlertoleranten Systemspeicher erfolgt eine Verlängerung der Zugriffszeit, falls vorher direkter Speicherzugriff erfolgte. Demgegenüber steht der Vorteil eines nebeneffektfreien Zugriffs, der eine der häufigsten Software-Fehlerursachen bei Assemblerprogrammierung praktisch beseitigt.
Die vorgestellte Lösung wird im Chemieanlagenbaukombinat Leipzig-Grimma serienmäßig eingesetzt.

Zusammenfassung

Ergänzend zu /6/ und /7/ wird eine weitere Methode angegeben, um Fehlertoleranzeigenschaften nicht nur in Arbeitsschritten, sondern auch nach Modul-Teilfunktionen realisieren zu können. Sie ist auf alle funktionell abgrenzbaren Diagnose-Objekte anwendbar, wenn die damit i. a. verbundene Verlängerung von Antwortzeiten vertretbar ist.
Die Methode wird am Beispiel einer Funktionseinheit mit Toleranz gegenüber Speicherfehlern erläutert.
Der Vorschlag erweitert die Palette nachrüstbarer Fehlertoleranz ohne wesentliche Eingriffe in Hardware und Betriebssystem. Die Fehlertoleranz trägt dazu bei, vorhandene Mikrorechner-Lösungen künftig höheren Zuverlässigkeitsanforderungen anpassen zu können.

***Tafel 1** Einfluß der Anzahl defekter Speicherzellen auf die Zugriffszeit /8/.*

Anzahl der bisher aufgetretenen Speicherfehler	*Faktor für Verlängerung der Zugriffszeit*
0	1
1	3
2	3,2
3	3,6
4	4,4
5	5
6	5,6
7	6,4
8	7,2

Literatur

/1/ Vogt, E.: Wege zur modernen Fabrik. Techn. Gemeinschaft (1987) 2, S. 15
/2/ Töpfer, H.: Bemerkungen zur Konzipierung hierarchisch strukturierter Automatisierungslösungen. messen, steuern, regeln 29 (1986) 8, S. 356
/3/ Kriesel, W.; Gibas, P.; Steinbock, K.: Neuartige Ansätze für die Technik künftiger Automatisierungsanlagen mit intelligenten Geräten. Tagungsmaterial der 5. Wiss. Konferenz „Anlagenautomatisierung", Leipzig, Wiss. Berichte (1986) 2, S. 45
/4/ Kopetz, H.: Konzepte zur Realisierung hochzuverlässiger Automatisierungssysteme. Fachberichte Messen Steuern Regeln Bd. 10, Interkama – Kongreß 1983, Springer-Verlag, Berlin (West) – Heidelberg – New York – Tokyo 1983
/5/ Weiss, R.: Fehlertolerante Rechnersysteme – Funktionsprinzipien u. Realisierungsformen. Regelungstechnische Praxis 25 (1983) 10, S. 408
/6/ Schäfer, M.: Fehlertolerante Mikrorechner-Strukturen/ Zuverlässigkeitsaspekt. Diss. A, TH Leipzig 1987
/7/ Kriesel, W.; Schäfer, M.: Fehlertolerante Mikrorechner-Funktionseinheiten mit der Redundanzart Graceful Degradation. Mikroprozessortechnik 2 (1988) 6, S. 165
/8/ Kirste, R.: Diagnose-Software für Einchipmikrorechner in fehlertoleranten Automatisierungslösungen. Dipl.-Arbeit, TH Leipzig, SAA 1987
/9/ Dal Cin, M.: Softwareimplementierte Fehlertoleranz. Informatik-Spektrum (1984) 7, S. 108
/10/ Parnas, D. L.: On the Criterias to be Used in Decomposing Systems into Moduls. Com. of the ACM. Vol. 15, No. 12, 1972

✉ KONTAKT ☏

Technische Hochschule Leipzig, Sektion Automatisierungsanlagen, Prof. Dr. sc. W. Kriesel, Postfach 66, Leipzig, 7030; Tel. 3 94 31 36

TERMINE

Fachtagung „Aufsetzmontage in der Leiterplattentechnik unter Berücksichtigung der Nacktchipverarbeitung"
WER? Fachverband Elektrotechnik in der Kammer der Technik, Wissenschaftliche Sektion Technologie des elektronischen Gerätebaues.
WANN? IV. Quartal 1988
WO? Berlin
WAS?
- Möglichkeiten und Grenzen der Baugruppenrealisierung
- Voraussetzungen zur Einführung der Aufsetzmontage für SMD
- Verfahren und Ausrüstungen
- Anwenderbeispiele
- Fachspezifische Ausstellung

WIE? Vortragsvorschläge und Ausstellungsangebote sowie Teilnahmemeldungen sind schriftlich an folgende Anschrift zu richten: Kammer der Technik – Präsidium – Fachverband Elektrotechnik, Clara-Zetkin-Str. 115/117, Berlin, 1086

Hoppe

dBASE-II-Datei nicht geschlossen – was nun?

Was niemendem passieren sollte ist, daß man eine Diskette mit einer dBASE-Datei aus dem Laufwerk entfernt, ohne die Datei vorher geschlossen zu haben.
Falls dies doch einmal passiert – was dann? Das ist ganz einfach. Wenn Sie irgendwann diese Datei wieder aufrufen, antwortet der Computer: ‚Not a dBase File'.
Die Daten sind futsch, könnte man denken. Irrtum! Power hilft! Und das geht auch noch sehr einfach! Sie rufen das Hilfsprogramm POWER auf und lokalisieren das Laufwerk, in dem sich die defekte Datei befindet. Geben Sie nun den Befehl TYPEX ein. Es erscheinen die Directory-Eintragungen mit der Frage: select ?. Hier geben Sie die Nummer Ihres Files ein und quittieren mit der ENTER-Taste.
Es wird Ihnen nun als erste Zeile die Position des ersten Blocks auf der Diskette angegeben. Sie merken sich die Werte von „T=????" und „S=???". Lassen Sie die Liste weiter laufen, so werden Sie sehen, daß Ihre Daten noch vorhanden sind. Mit ESC (Escape) können Sie diese Liste abbrechen.
Nun kehren Sie zurück ins dBASE und erstellen eine Datei anderen Namens mit der gleichen Struktur. Diese Datei kann am Ende der nun folgenden Prozedur wieder gelöscht werden.
Sie laden nun wieder das POWER, geben nochmals den Befehl TYPEX und listen jetzt die neu erstellte Datei. Wiederum merken Sie sich die Werte der Position des ersten Blocks („T=????" und „S=???").
Anschließend lesen Sie den ersten Block der neu erstellten Datei in den RAM-Speicher Ihres Computers. Der Befehl lautet:
READ T S XX.
Ein Beispiel mit Werten:

T=0D00, S=012
Befehl READ D00 12 XX

Nun haben Sie den ersten Block Ihrer neu erstellten Datei auf dem Monitor und können mit dem Befehl WRITE T S diesen Block auf die defekte Datei kopieren. Hier verwenden Sie nun die Position des ersten Blocks der defekten Datei.
Wieder ein Beispiel:

T=0047, S=049
Befehl WRITE 47 49

Ist dieser Blocktransfer gelungen, so laden Sie DBASE und rufen Sie mit USE Ihre Datei auf. Sie werden feststellen, daß Ihre Daten noch vollständig vorhanden sind. Viel Erfolg!

U. Zielinski

Kleines Lexikon zur Fehlertoleranz

zusammengestellt von Prof. Dr. Kriesel



Fehlertoleranz

Diversität
ist die Verwendung unterschiedlicher Komponenten gleicher Funktionalität. D. bedeutet z. B. die redundante Implementierung von Nutzfunktionen durch mehrere verschiedenartig (zweites Team, andere Programmiersprache o. ä.) entworfene Subsysteme.

Fehlerdiagnose
Das Ziel der Fehlerdiagnose ist es, die Fehlerursachen in angebloser Zeit und mit angebbarer Wahrscheinlichkeit qualitativ und quantitativ zu erkennen (Fehlererkennung) und den Fehlerort (z. B. betroffene Module) einzugrenzen (Fehlerlokalisierung).

Fehlertoleranz
ist die Eigenschaft eines Systems, auch bei Vorhandensein interner Fehler in einzelnen Komponenten nach außen die spezifizierten Funktionen korrekt auszuführen.

Graceful Degradation
(sanfter Leistungsabfall, definierte Leistungsminderung).
Im Fehlerfall werden bestimmte Teilfunktionen eines Systems zurückgestellt, um seinen Totalausfall zu vermeiden.
Das System funktioniert damit in einem Zustand verminderter Leistungsfähigkeit, der vorübergehend tolerierbar sein muß.

Majoritätssystem (Auswahlsystem)
ist ein redundantes System, das nur dann funktioniert, wenn von n insgesamt vorhandenen Komponenten (Teilsystemen) mindestens r funktionieren. Zum Beispiel kann ein 3fach redundant ausgelegtes System noch als funktionsfähig gelten, wenn mindestens zwei Teilsysteme noch einwandfrei arbeiten, die über ein Mehrheits-Auswahlsystem herausgefunden werden (2-von-3-System).

Redundanz
ist ein funktionsbereites Vorhandensein von mehr als für die vorgesehene Funktion notwendigen technischen Mitteln, die im Bedarfsfall zur Funktionssicherung mit herangezogen werden („Weitschweifigkeit“).

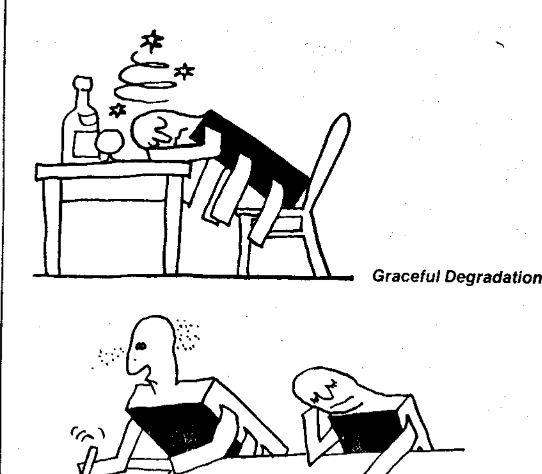
Rekonfiguration
ist die Fähigkeit eines fehlertoleranten Systems, aufgrund eines Diagnose-Ergebnisses seine Struktur zu ändern.

Stand-by-System
ist eine Anordnung mit einem Reservesystem, das erst bei Ausfall des Hauptsystems eingeschaltet (aktiviert) wird, davor jedoch praktisch nicht belastet wurde. Hierbei tritt eine Unterbrechungszeit vom Erfassen des Ausfalls bis zur Beendigung des Umschaltens ein.

Verfügbarkeit
ist die Wahrscheinlichkeit dafür, daß ein System zum Zeitpunkt t intakt ist.

Wiederanlauf (Recovery)
ist die Fortsetzung der Arbeit eines Systems nach erfolgter Strukturänderung unter definierten Bedingungen und mit aktuell gültigen Informationen.

Karikaturen: *Eggstein (4)*



Graceful Degradation

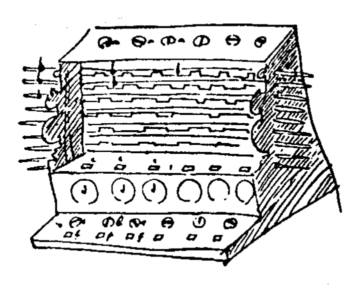
Stand-by-System

Wegbereiter der Informatik



WILHELM SCHICKARD

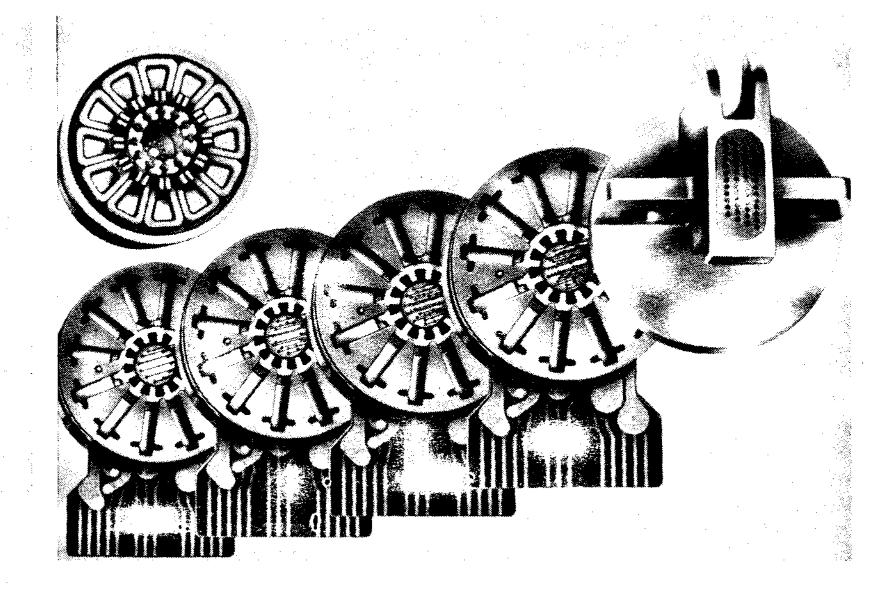
*** 1592 Herrenberg, † 1635 Tübingen.**



Nach Wiederentdeckung (1957) zuverlässiger Quellen ist nunmehr mit Sicherheit erwiesen, daß bereits im Geburtsjahr Pascals, also 1623, in Tübingen eine Rechenmaschine entwickelt worden ist, und zwar von dem Tübinger Universitätsprofessor Wilhelm Schickard (auch die Schreibweisen Schickart oder Schickhardt sind gebräuchlich).
Schickard war ein außerordentlich vielseitiger Gelehrter und galt nicht nur als einer der bedeutendsten Hebraisten und Orientalisten seiner Zeit, sondern er hat sich auch als Astronom, Mathematiker, Geodät, Kupferstecher und Maler hervorgetan. Jedoch ursprünglich war er Theologe, und als solcher wurde er 1611 schon Magister und Repetitor am Tübinger Stift, 1614 Diakon und ab 1619 Professor für Hebräisch, Aramäisch und weitere orientalische Sprachen. Ebenso wie sein bekannterer Onkel, der Baumeister Heinrich Schickard, war er zudem technisch und mathematisch hochbegabt. Im Jahre 1631 wurde er Professor für Mathematik und Astronomie und trat damit die Amtsnachfolge des verstorbenen Mathematikers und Astronomen M. Maestlin an, der Keplers Lehrer gewesen war. In dieser Amtszeit hat er verschiedene astronomische Geräte erfunden.
Schickard betätigte sich aber auch als Geodät. So führte er die erste Landesaufnahme von Württemberg nach eigenen kartographischen Methoden durch; diese Methoden hat er in einem Buch dokumentiert, das nach seinem Tode noch oft aufgelegt worden ist. Des weiteren hat er mit mathematischen Mitteln das sogenannte Pothenotsche Problem gelöst, ein von W. Snellius aufgestelltes, später nach dem französischen Mathematiker Laurent Pothenot (1660–1732) benanntes geodätisches Ortsbestimmungsproblem.
Seine Rechenmaschine baute Schikkard auf Anregung des befreundeten Johannes Kepler, der ja für seine astronomischen Arbeiten viele arithmetische Operationen auszuführen hatte und an einer maschinellen Erleichterung des Rechnens interessiert gewesen sein dürfte. Zur Durchführung eines solchen Vorhabens mußte Schickard erst das Ziffernrad und die Zehnerübertragung erfinden.
Seine Maschine ermöglicht neben Addition und Subtraktion auch Multiplikation und Division; es lassen sich 6stellige Operanden einstellen, das Resultatregister ist 8stellig. Es ist belegt, daß diese Maschine gut funktioniert hat. Das weiß man aus Sofort-Briefen des Erbauers an Kepler und geht auch daraus hervor, daß 1624 in Tübingen ein zweites Exemplar für Kepler hergestellt wurde, aber mit dem Hause des Mechanikers Pfister verbrannt ist.
Glücklicherweise sind Schickards technische Beschreibungen (eben in Briefen an Kepler) und Anweisungen für seinen Mechaniker sowie einige Skizzen (eine davon zeigt unser Bild) erhalten geblieben und so ausreichend detailliert, daß nach diesen Angaben seine Maschine in der Neuzeit nachgebaut werden konnte. So steht nun seit 1960 ein Exemplar im Tübinger Rathaus, weitere Modelle davon befinden sich u. a. in der Geburtsstadt Herrenberg, im Deutschen Museum München sowie in der Sammlung von IBM in New York.

Dr. Klaus Biener



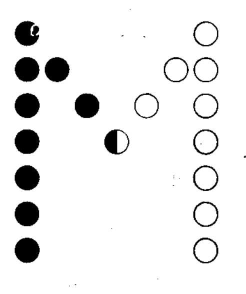



Erster 48-Nadel-Drucker

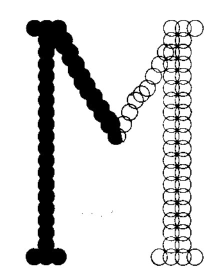
Zur CeBit '88 in Hannover stellte die japanische Firma EPSON den ersten 48-Nadel-Drucker der Welt vor. Der Matrixdrucker verbindet die Universalität der Impact-(Anschlag-)Technik mit einer Druckqualität, die es bisher nur bei Non-Impact-Druckern gab (anschlaglose Drucker, z. B. Laser- oder Tintenstrahldrucker). Die bewährten 9-Nadel-Drucker, die ständig verbessert wurden und werden und heute beachtliche Leistungen erzielen oder die 24-Nadel-Drucker wird der „48-Nadler" wohl nicht ersetzen, da er aufgrund des höheren Konstruktionsaufwandes wesentlich teurer ist. Für bestimmte Anwendungen wie Textverarbeitung, CAD oder Desktop Publishing (DTP) bietet er jedoch beste Voraussetzungen, da die Auflösung von 360 × 360 Punkten/Zoll (das ergibt etwa 200 Punkte je Quadratmillimeter) noch über der vieler Laserdrucker liegt.
Die Nadeln sind in vier Reihen zu je 12 angeordnet und um 1/360 Zoll untereinander versetzt. Die Steuerelektronik feuert die Nadeln in einem Takt ab, der auch in horizontaler Richtung für einen Punkteabstand von 1/360 Zoll sorgt. Dadurch besitzt der 48-Nadel-Drucker die volle Auflösung von 360 × 360 Punkten in einem einzigen Durchgang. Bei 24-Nadel-Druckern ist hierfür ein zweiter Druckvorgang mit versetzten Punkten, ähnlich dem NLQ-Modus bei 9-Nadel-Druckern, erforderlich (NLQ – near letter quality, nahe Briefqualität). Die Druckgeschwindigkeit des 48-Nadel-Druckers beträgt maximal 300 Zeichen pro Sekunde, in Schönschrift sind es 100 cps. Die Buchstabenmatrix wird bei Schönschrift in der vollen Auflösung von 360 × 360 Punkten pro Zoll wiedergegeben. Der beim Drucken anfallende Datendurchsatz ist im Vergleich zum 24-Nadel-Drucker mehrfach höher. Er entspricht bei einer Geschwindigkeit von 300 cps derjenigen eines 24-Nadel-Druckers, der mit 600 cps arbeitet. Um die Datenflut für 48 Nadeln bei hoher Druckgeschwindigkeit zu bewältigen, ist der 48-Nadel-Drucker als echtes Multiprozessorensystem aufgebaut. Für die Codierung und den Datenfluß wird eine 16-Bit-CPU, für die Steuerung der Mechanik (Druckkopf und Papiermanagement) zusätzlich eine 8-Bit-CPU eingesetzt. Der 48-Nadler benötigt keine besondere Steuersoftware, sondern bietet bei der Wiedergabe von Text die Druckqualität auch mit der vorhandenen Software. Dafür sorgt das Betriebssystem, das alle Zeichen in der vollen Auflösung im ROM gespeichert hat. Für die Wiedergabe von Grafik arbeitet der 48-Nadler auch mit den vorhandenen 24-Nadel-Treibern. Für das volle Ausschöpfen der Druckqualität im Grafikbetrieb sind entsprechende Treiber für 48 Nadeln einzusetzen.
Der Drucker führt alle Papierfunktionen vom Einzug bis zum wechselseitigen Zustellen von Einzelblatt und Endlospapier automatisch durch. Beide Papierarten können abwechselnd bedruckt werden, ohne daß der Einzelblatteinzug entfernt werden muß. Es können ein Original und bis zu drei Durchschläge gedruckt werden. Ebenso können Briefumschläge automatisch zugestellt werden. Mit Hilfe einer Micro-Feed-Funktion läßt sich jede Papierart millimetergenau positionieren. Das Problem der richtigen Druckkopfeinstellung bei Verwendung unterschiedlicher Papiersorten im Wechsel mit Formularsätzen ist ebenfalls durch eine Automatik gelöst: Der 48-Nadel-Drucker stellt sich selbsttätig auf die Papierdicke ein. Von Haus aus ist er als Drucker im Breitformat (A3) ausgelegt. Damit wird seiner Eignung für alle Einsatzgebiete, auch Desktop-Publishing und CAD, entsprochen. Standardmäßig verfügt der 48-Nadel-Drucker über eine parallele Schnittstelle sowie über das EPSON-Betriebssystem ESC/P, das inzwischen auf dem Druckermarkt als Industriestandard anerkannt ist. Es ermöglicht eine Zeichensatz- und Steuercodeanpassung praktisch an jedes PC-System. Als Option sind RS 232C, IEEE 488 und die 81xx-Serie lieferbar.

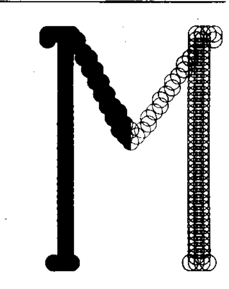
Fotos und Grafik: EPSON

Das Bild zeigt schematisch einen auseinandergenommenen 48-Nadel-Druckkopf. Er besteht aus vier hintereinander liegenden Führungseinheiten mit Magneten, von denen jede über 12 Nadeln und die zugehörige Elektronik und Mechanik verfügt. Die Nadeln werden in Bohrungen durch die Führungseinheiten nach vorne durchgeführt. Im hintersten Ring ganz links sind es 12, einen Ring weiter bereits 24, dann 36 und schließlich 48 Nadeln. An der Spitze des Druckkopfes treten sie in vier nebeneinanderliegenden Reihen aus (ganz rechts). Die hinterste Führungseinheit ist auch von der Rückseite gezeigt, so daß man den Aufbau erkennen kann: Dünne Bohrungen für die Nadeln, Bohrungen für Spiralfedern und 12 Elektromagnete sind ringförmig um den Mittelpunkt angeordnet.



9 Nadel-Matrix


24 Nadel-Matrix


48 Nadel-Matrix

Die Grafik zeigt das Schriftbild, das 9-, 24- und 48-Nadel-Drucker nach einem Durchgang erzeugen können. Das „M" links ist in der 9-Punkte-Matrix wiedergegeben. Ein geschlossenes Schriftbild (Near Letter Quality) wird durch zweimaliges Überfahren erzeugt. 24-Nadel-Drucker (Bildmitte) benötigen nur einen Durchgang und erzeugen dennoch ein geschlossenes Schriftbild (Letter Quality). Sie besitzen zwei Reihen von je 12 Nadeln, die in etwa 0,85 mm Abstand parallel verlaufen. Der Druckkopf feuert zunächst die ersten 12 Nadeln ab, fährt dann 0,85 mm weiter und setzt die zweiten 12 Nadeln genau zwischen die Abdrücke der ersten 12. So ergibt sich eine durchgehende Linie. Nochmals doppelt so dicht sitzen die Punkte beim 48-Nadel-Drucker (das „M" rechts). Hier werden kurz nacheinander vier mal 12 Nadeln abgefeuert. Die Punkt-zu-Punkt-Abstände betragen in jeder Richtung nur noch 1/360 Zoll, das entspricht 0,07 mm. Dabei überschneiden sich die Punkte zu etwa 90 Prozent, so daß sich eine sehr geschlossene, gleichzeitig klare und scharfe Schrift ergibt.

Rechnerinterface zur Eingabe und Korrektur von Sensordaten

Dr. Michael Bratge
Forschungszentrum des Kombinates VEB Carl Zeiss JENA

Mit der byteseriellen Dateneingabe und -verarbeitung bei hoher Bitrate erreicht ein Mikrorechner schnell die Grenze seines Leistungsvermögens. Ein Beispiel hierfür ist die Übernahme und Korrektur von Daten aus CCD-Zeilen. Zur Lösung dieser Aufgabe wurde eine spezielle Interfacebaugruppe (Sensorinterface) für das MMS16 entwikkelt, die darüber hinaus geeignet ist, eine Vielzahl von Sensorsignalen, die digitalisiert vorliegen, zu korrigieren und/oder zu klassieren.

Hardware

Das MMS16 ist ein 16-Bit-Multimikrorechnersystem mit mehreren Bussen (Bild 1). Ein Systembus kann als Kommunikationsmittel zwischen mehreren Busmastern (z. B. ZVE) dienen, die ihrerseits wiederum eigene Residentbusse bilden, über welche sie mit ihren Slave-einheiten (I/O-, Speichereinheiten) arbeiten.
Das Sensorinterface ist ein intelligenter Slave am Systembus des MMS16. Es unterteilt sich in 2 Funktionseinheiten, die auf je einer Steckeinheit untergebracht sind: ein Operationswerk und ein Steuerwerk. Beide sind über einen Bussteckverbinder und eine Busleiterplatte des MMS16 miteinander verbunden, womit das Sensorinterface seinen privaten Residentbus realisiert. Über diesen Bus erfolgt auch die Verbindung mit der Eingangsdatenquelle. Die Kontaktbelegung ist so gewählt, daß eine Busanzeige des MMS16 auch zur Anzeige dieses Busses genutzt werden kann. Das Sensorinterface adressiert insgesamt 16 KByte RAM, auf den jeder Master des Systems über Systembus zugreifen kann, wenn er das Interface programmiert, steuert oder mit ihm Daten austauscht.

Operationswerk

Das Blockschaltbild des Operationswerkes zeigt Bild 2. Das Operationswerk besteht in seinem Kernstück aus einer arithmetisch-logischen Verarbeitungseinheit (ALU), an deren Eingang A die Eingangsdaten oder rückgekoppelte Zwischenergebnisse mit einer Breite von maximal 12 Bit geführt werden.
Eingang B ist mit den Datenleitungen eines zu 5K × 16 Bit organisierten Operationsspeichers verbunden. Dieser Speicher ist in 2 Bereiche unterteilt, einen Korrekturwertspeicher (4K × 16 Bit) und einen Ergebnisspeicher (1K × 16 Bit). Der Korrekturwertspeicher wird durch einen Zähler angesteuert, der Ergebnisspeicher durch einen zweiten Zähler und über die zwei höchstwertigen Adreßleitungen direkt durch das Mikroprogrammsteuerwerk. Mit Hilfe dieser zwei Adreßleitungen lassen sich weitgehend unabhängig voneinander bis zu vier verschiedene Ergebnisfelder aufbauen. Auf den Operationsspeicher kann ebenfalls über den Systembus des MMS16 zugegriffen werden.
Der Ausgang der ALU führt auf zwei EPROMs, die als Transformationsstufe fungieren, welche im Anwendungsbeipiel das Sensor-Interface logarithmieren bzw. potenzieren kann. Parallel dazu liegt ein Datenweg, der genutzt wird, wenn die Daten nicht transformiert werden sollen. Vom Ausgang des Transformationsgliedes können die Daten über ein Register FB auf den Eingang A der ALU rückgekoppelt oder in ein Zwischenregister P geschrieben werden, welches die Daten in den Operationsspeicher übergeben kann.
Die ALU ist 12 Bit breit organisiert, das Carry-Ausgangssignal ist auf den Zähleingang eines Zählers geführt, dessen Dateneingänge mit den vier höchstwertigen Datenleitungen des Operationsspeichers verbunden und dessen Datenausgänge auf die vier höchstwertigen Dateneingangsleitungen der Register P und FB geführt sind. Die Zählerstände werden mit den Ergebnsidaten der ALU in das entsprechende Register geladen. Auf diese Weise ist es möglich, Additionsoperationen bis zu 16 Bit Breite, allerdings ohne nachfolgende Transformation, auszuführen. Außerdem kann der Zähler für Zähloperationen genutzt werden.
Das FB-Register ist ein Schieberegister, das in der Lage ist, den gespeicherten Wert nach rechts zu verschieben und von links mit Nullen aufzufüllen, somit jeweils durch 2 zu dividieren.
Die zwei höchstwertigen Datenbits (Bit 14, 15) des Korrekturwertspeichers lassen sich als Bedingungsflags für Verzweigungen im Mikroprogramm nutzen.
Der Transformations-EPROM kann nur 12 Bit breite Werte verarbeiten. Damit bieten sich die vier höchstwertigen Ausgänge der EPROMs an, um auf vier Ladeeingänge des Ergebnisspeicherzählers geführt zu werden. Damit ist die Baugruppe in die Lage versetzt, Meßwerte in bis zu 16 Klassen zu klassieren, wobei jeder Meßwert einer frei wählbaren Klasse zugeordnet werden kann. Jede Klasse wird somit durch einen Speicherplatz des Ergebnisspeichers repräsentiert, dessen Inhalt erhöht wird, sobald ein Meßwert in die Klasse fällt.
Die Eingänge des den Korrekturwertspeicher steuernden Zählers sind auf einen Kopfsteckverbinder geführt, so daß ein beliebiger Anfangskorrekturwert extern ausgewählt werden kann. Dies empfiehlt sich, wenn dauernd unterschiedliche Bereiche eines mehrdimensionalen Sensors, beispielsweise einer CCD-Zeile, verwendet werden.

Steuerwerk

Das Blockschaltbild des Steuerwerkes zeigt Bild 3. Das Steuerwerk basiert auf einem Mikroprogrammspeicher aus einem zu 1K mal 32 Bit organisierten RAM, auf den über den Systembus des MMS16 zugegriffen werden kann. Über den Systembus erfolgt auch die Mikroprogrammierung. 10 der 32 Datenbits wählen die Folgeadresse des Mikroprogramms aus, wovon 8 Bit unmittelbar Adreßbit der Folgeadresse bilden und 2 Bit von außen gesetzte Flags zurücksetzen können, die auf 2 weitere Adreßbit der Folgeadresse wirken.

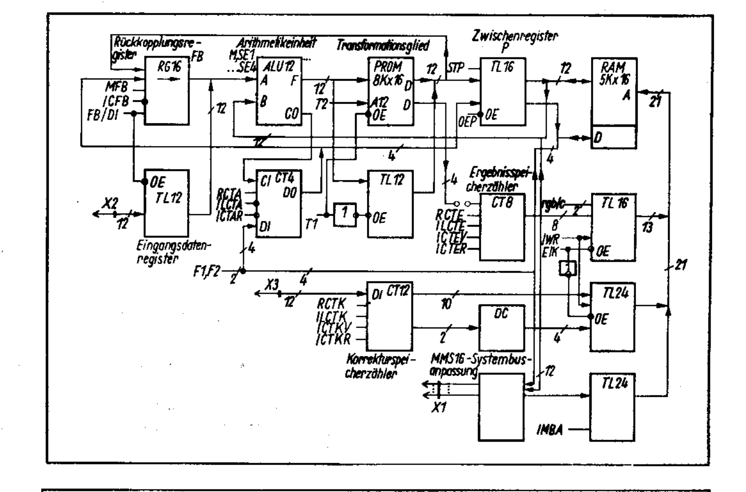


Bild 2 Operationswerk

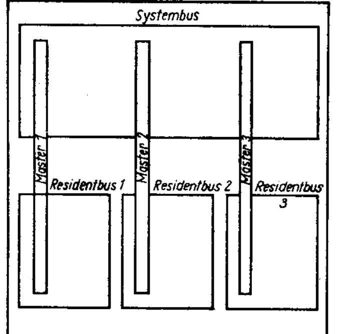


Bild 1 MMS16-Bussystem

Bild 3 Steuerwerk

Der Mikroprogrammspeicher umfaßt 1 K Doppelworte (DWorte) und wird in vier Bereiche unterteilt, je nachdem, welche Flags (F1, F2) gesetzt bzw. rückgesetzt sind. Im Normalfall läuft das Programm im Bereich mit rückgesetzten Flags. Wird ein Flag gesetzt (Eingangsdatengültigkeitssignal oder gesetztes Flag im Operationsspeicher, wenn Datenbit 14 bzw. 15 des Operationsspeichers = 1), springt das Mikroprogramm in den entsprechenden Bereich. Die Grundadresse (Bit 0 ... 7) bleibt hierbei unverändert.

Das Steuerwerk wird durch einen 16-MHz-Taktgenerator in Verbindung mit einem Schieberegister, welches wahlweise 2 ... 8 Taktphasen (P1 ... P8) erzeugen kann, gesteuert. An die Taktphasen werden über ein Wickelfeld Steuersignale für das Steuerwerk angeschlossen.

Der Zugriff auf Operations- und Mikroprogrammspeicher wird über einen Parallelinterfacebaustein K580 WW55A (PPI) gesteuert. Über eine weitere Datenleitung dieses Bausteins wird das Mikroprogramm durch einen Master des MMS16 gestartet bzw. rückgesetzt. Die restlichen Datenleitungen des PPI sind mit den Dateneingangsleitungen und deren Gültigkeitssignalen verbunden. Damit ist ein kompletter Selbsttest der Baugruppe mit Hilfe eines Masters des MMS16 möglich. Die Eingangsdaten werden hierbei über das PPI simuliert.

Mikroprogrammierung

Die Mikroprogrammierung erfolgt, indem jeder Mikroprogrammbefehl durch einen Master des MMS16 in den Mikroprogrammspeicher eingeschrieben wird. Da das sehr umständlich ist, wurde mit Hilfe der MACRO-UTILITY des verwendeten Assemblers ein Satz Mikroprogrammbefehle erstellt. Mit Hilfe dieser MACROs kann ein Mikroprogramm bequem geschrieben werden, ohne daß der Programmierer wissen muß, welches Bit im Mikroprogrammspeicher er für welches Steuersignal setzen muß. Die Mikroprogrammbefehle unterteilen sich in:

1. Initialisierungsbefehle,
die das Operationswerk in einen definierten Anfangszustand versetzen,

2. Hauptbefehle, die
- Quellregister auswählen,
- die ALU bzw.
- das Transformationsglied einstellen oder
- Zielregister (P, FB) auswählen,

3. Komplexe Hauptbefehle,
die in geeigneter Weise mehrere der 4 vorstehenden Gruppen von Hauptbefehlen kombinieren,

4. Zusatzbefehle,
- Speicherauswahl und -schreibbefehle
- Interruptbefehle, die einen INTERRUPT an einen Master des MMS16 abgeben,
- Befehle zum Rücksetzen der Flags,

5. Abschlußbefehle,
die die Folgeadresse des Mikroprogramms eintragen und den nächsten Mikroprogrammbefehl auswählen.

Ein vollständiger Mikroprogrammbefehl besteht dabei immer aus mehreren Mikroprogrammbefehlen oben genannter Art, deren letzter stets ein Abschlußbefehl ist. Mit einem entsprechenden Rahmen wird aus den Mikroprogrammbefehlen ein Assembler-Unterprogramm, bei dessen Aufruf das Mikroprogramm in den Mikroprogrammspeicher geladen wird. Der Mikroprogrammbefehlssatz kann bei Bedarf, insbesondere bei Notwendigkeit der Nutzung weiterer ALU-Befehle, leicht erweitert werden.

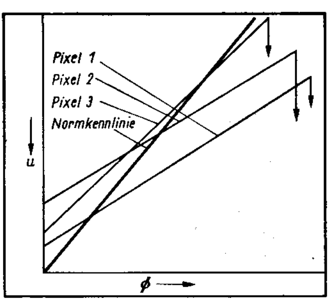


***Bild 4** Unkorrigierte Kennlinien der Pixel von CCD-Zeilen. Die Pfeile geben die notwendige Verschiebung der Kennlinien für die Dunkelkorrektur an.*

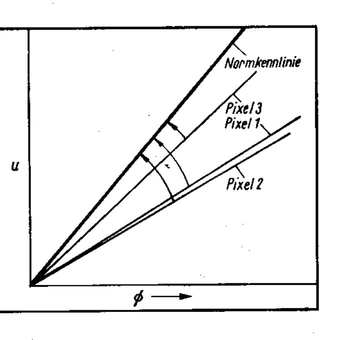


***Bild 5** Dunkelkorrigierte Kennlinien der Pixel von CCD-Zeilen. Die Pfeile geben die notwendige Drehung der Kennlinien für die Hellkorrektur an.*

Anwendungsbeispiel

Die Kennlinien der Pixel einer CCD-Zeile sind, bedingt durch unterschiedlichen Dunkelstrom und Empfindlichkeit, alle unterschiedlich (Bild 4). Zur Weiterverarbeitung der Meßwerte sind die Kennlinien der Pixelelemente untereinander zu normieren. Weiterhin ist über eine bestimmte Anzahl von Meßwerten der Mittelwert zu bilden. Diese Anzahl ist eine Potenz von zwei. Zeitmultiplex sind die Daten aus drei CCD-Zeilen zu verarbeiten, wodurch eine Zeit von 6 Mikrosekunden als Vorgabe für die Verarbeitung eines Meßwertes resultiert.

Dieser Prozeß soll durch das Sensorinterface realisiert werden. In einem ersten Schritt, der Dunkelkorrektur, wird der Dunkelkorrekturwert vom Meßwert subtrahiert (Bild 4 und 5).

Durch eine anschließende Multiplikation jedes Meßwertes mit einem ihm zugeordneten Hellkorrekturwert wird eine Hellkorrektur durchgeführt, das heißt die unterschiedliche Empfindlichkeit aller Pixel korrigiert. Das Sensorinterface realisiert diese Multiplikation durch Logarithmierung, anschließende Addition und Entlogarithmierung.

Die Dunkel- und Hellkorrekturwerte lassen sich unter Zuhilfenahme des Sensorinterfaces in einem Eichvorgang ermitteln. Hierzu wird das Sensorinterface mit je einem entsprechenden Mikroprogramm geladen.

Die Korrekturwerte werden in der Reihenfolge, in der die Daten der CCD-Zeilen ankommen, wortweise im Korrekturwertspeicher abgelegt, und zwar immer ein Dunkelkorrekturwert und anschließend der dazugehörige Hellkorrekturwert.

Das Sensorinterface führt daraufhin im Normalberieb folgende Schritte aus:

1. Dunkelkorrektur

Subtraktion des Dunkelkorrekturwertes vom Meßwert, anschließendes Logarihmieren und Schreiben in FB-Register.

2. Hellkorrektur

Addition des Hellkorrekturwertes zum dunkelkorrigierenden Meßwert, anschließendes Entlogarithmieren und Schreiben in FB-Register. (Der erste Wert einer zu mittelnden Meßwertgruppe wird nicht in das FB-Register, sondern in den Ergebnisspeicher geschrieben.)

3. Mittelwertbildung

Addition des korrigierten Meßwertes zum Wert im Ergebnisspeicher. Damit entsteht im Ergebnisspeicher die Summe der korrigierten Meßwerte einer zu mittelnden Gruppe. Nach der Addition des letzten Wertes wird die im Ergebnisspeicher stehende Summe ins FB-Register geladen und dort nach rechts verschoben – somit durch die Anzahl der Meßwerte dividiert. Danach wird der Mittelwert logarithmiert und wieder in den Ergebnisspeicher geschrieben.

In Abhängigkeit von der Lage des jeweiligen Eingangswertes im zu mittelnden Feld (Feldanfang, -ende) benötigt das Sensorinterface für dessen Verarbeitung zwischen 1,8 und 3,4 Mikrosekunden.

Literatur

/1/ Hoffmann, Rolf: Rechenwerke und Mikroprogrammierung. München, Wien, Oldenburg 1977

/2/ Bratge, M.; Wagner, H.; Schnetter, R.: Korrektureinrichtung für Signale von Meßgliedern. WP-Anmeldung G01D/240 822 7.

⊠ KONTAKT ☏

Kombinat VEB Carl Zeiss JENA, Forschungszentrum, Abt. WIR 2, Carl-Zeiss-Straße 1, Jena, 6900; Tel. 83 34 19

TERMINE

Internationales Problemseminar „Compiler Compiler and High Speed Compilation"

WER? AdW, Institut für Informatik und Rechentechnik
WANN? 10.–14. Oktober 1988
WO? Berlin
WAS?
1. Moderne und schnelle Compilerterchniken
2. Methoden und Werkzeuge für die Generierung von Compilerkomponenten
3. Zwischensprachliche Repräsentationen
4. Werkzeuge für Programmierumgebungen, z. B. Editor und Debugger

WIE? Teilnahmemeldungen und Vortragsmeldungen (mit Kurzfassung) an Akademie der Wissenschaften der DDR, Institut für informatik und Rechentechnik, M. Albinus, Rudower Chaussee 5, Berlin, 1199

M. Albinus

Fachtagung „EAW-Elektronik"

WER? Bezirksvorstand der KDT Frankfurt (O.)
WANN? 23./24. 11. 1988
WO? Frankfurt (O.)
WAS?
P 8000, S 2000 und weitere Geräte des KEAW
WIE? Nähere Informationen sind bei der BO Frankfurt (O.) der KDT, Ebertusstr. 2, Frankfurt (O.), 1234, Tel. 36 93 60 erhältlich.

Georgi

Ein C-Compiler für den KC 85/1

Dr. Reinhard Wobst, Dresden

Im folgenden wird ein Einpaßcompiler für die Sprache C vorgestellt. Der Compiler Pretty C ist etwa 19 KByte lang, verarbeitet fast die gesamte Sprache und ist in seiner Arbeitsweise etwas an TURBO-PASCAL angelehnt.

Allgemeines

Wegen der Leistungsfähigkeit der Sprache C und vor allem wegen ihrer sehr guten Übertragbarkeit auf andere Rechner ist ein C-Compiler auch auf dem KC 85/1 von großem Nutzen.

Beim KC 85/1 gibt es aber eine einschneidende Randbedingung: Als externer Speicher steht nur das Magnetband zur Verfügung. Die sonst übliche Arbeit mit einem Linker bzw. mit mehreren Pässen bei der Compilierung würde damit viel zu umständlich werden. Der einzige dem Autor bekannte C-Compiler, der nur mit Magnetband arbeitet, ist Hisoft-C für ZX Spectrum. Dieser Compiler ist jedoch kaum übertragbar und verarbeitet u. a. nur einen Teil der Datentypen in C. Daher entstand die Notwendigkeit, selbst einen Compiler zu schreiben.

Pretty C ist in Assemblersprache geschrieben und arbeitet auf Rechnern mit mindestens 32 KByte RAM, besser aber 48 KByte.

Schwerpunkte bei der Entwicklung waren:

● gute Speicherausnutzung, u. a.

– sind die meisten Listen gegeneinander verschieblich und ohne Längenbegrenzung

– enthält der generierte Code zahlreiche Optimierungen wie: Auflösen konstanter Teilausdrücke zur Compilierzeit, gesonderte Behandlung konstanter Feldindizes, Unterdrükken des Ladens nicht benötigter Teilergebnisse in Stackzellen

– werden zum Adressieren dynamischer Variabler nur 3 Bytes benötigt (d. h. soviel wie bei statischen Variablen), wodurch ihre Verwendung nicht nur bei Rekursionen sinnvoll wird.

● komfortable Nutzung (daher der Name „Pretty C")

● möglichst vollständige Implementierung der Sprache.

Im Laufzeitverhalten ist Pretty C weniger gut.

Sprachumfang

Wichtige Spracheinschränkungen sind:

– Bitfelder und der ‚enum'-Typ sind nicht implementiert

– Initialisierungen sind nicht zulässig (außer der automatischen Initialisierung aller statischen Variablen mit 0).

– Die Speicherklasse von Funktionen wird nicht berücksichtigt, insbesondere ist ‚static' wirkungslos.

– Der Präprozessor versteht nur die Kommandos #define und #undef.

Alle anderen Sprachelemente, beispielsweise Deklarationen innerhalb verschachtelter Blöcke, Gleitkommazahlen, ‚void' und ‚unsigned long' verarbeitet Pretty C wie in /1/ beschrieben.

Erweiterungen sind im wesentlichen:

– Keine automatische Konvertierung von ‚float' auf ‚double', sondern nur in binären Operationen, wenn einer der Operanden vom Typ ‚double' ist.

Pretty C enthält zwei getrennte Gleitkommaarithmetiken: 4-Byte-Zahlen für ‚float' und 8-Byte-Zahlen für ‚double'. Diese Trennung ist angesichts des großen Rechenzeitunterschieds zwischen einfacher und doppelter Genauigkeit auf dem U 880 wichtig.

– Eine inline-Anweisung, um Maschinencode direkt im Quelltext unterzubringen.

Arbeitsweise

Pretty C arbeitet mit Textpuffern, die über Namen angesprochen, erzeugt und gelöscht werden, sowie mit Files auf Magnetband. Diese Puffer bzw. Files werden über den im Compiler integrierten Editor verwaltet. Dieser Editor ist aus Platzgründen nur zeilenorientiert, aber relativ komfortabel und einfach zu bedienen. Quelltext kann auf beliebig viele Puffer/Files verteilt sein, wobei jeder Puffer bzw. File als File im Sinne von C betrachtet wird (Datenverkehr zwischen ihnen mittels externer Namen).

Pretty C übersetzt immer nur einen Puffer/File in einem Lauf. Die einzelnen Übersetzungsläufe werden dabei logisch verbunden. Mit Beendigung der Übersetzungen steht ein abarbeitsfähiges Programm im Speicher.

Vor allem ist es auch möglich, Files sofort beim Einlesen ohne Zwischenspeicherung zu übersetzen und damit viel Platz zu sparen.

Die Steuerung des Compilers geschieht vom Monitor aus über Kommandozeilen mit Schaltern, z. B.: > CC /DS

(Anzeigen noch nicht definierter, aber deklarierter Funktionen und externer Variabler, der Codelänge, Optionen usw.). Bei mehreren, verbundenen Übersetzungsläufen schreibt man zweckmäßig die einzelnen Kommandos in einen Textpuffer und läßt diesen vom Pretty-C-Kommandoprozessor abarbeiten.

Fehlersuche und Quelltextdebugger

Die schnelle und komfortable Fehlersuche ist einer der wichtigsten Vorzüge von Pretty C.

Fehlermeldungen erscheinen prinzipiell im Klartext. Syntaxfehler werden mit genauer Lokalisierung im Quelltext angezeigt. Danach bricht der Compiler ab, denn die Folgefehler sind meist unsinnig. Diese schrittweise Korrektur ist aber nicht störend, weil Pretty C recht schnell übersetzt (etwa 1 KByte Quelltext je Sekunde) und man nach einer Fehlermeldung mit nur wenigen Tastenbetätigungen im Editor und an der Stelle ist, wo der Fehler erkannt wurde. Zur Laufzeit werden auch Arithmetikfehler erkannt (was leider viele Laufzeitsysteme nicht können). Zusätzlich zum Fehlertext erscheint eine backtrace-Liste, d. h., die Kette der gerufenen Funktionen wird ausgegeben. Mit einem speziellen Lauf des Compilers kann der Fehlerort ebenso präzise wie bei Syntaxfehlern ermittelt werden.

Einige Funktionen von Pretty C helfen außerdem, Fehler zu vermeiden:

– Unterstützen des Editors beim Schreiben von Schlüsselwörtern (weniger Tippfehler!)

– Unterstützen des Editors beim Strukturieren des Quelltextes

– automatisches Einrücken im Listing des Compilers

– Überprüfen der Argumentezahl von Funktionen zur Laufzeit

– Erkennen nicht definierter Funktionen

– abschaltbare Fehleranzeige beim Zugriff auf Werte außerhalb der Datenbereiche (über Pointer oder Felder) – die Hauptursache für Abstürze.

Das wichtigste Hilfsmittel bei der Fehlersuche ist jedoch ein Quelltextdebugger, mit dem ein Programm schritt- oder abschnittsweise abgearbeitet werden kann bei gleichzeitiger Anzeige des Quelltextes. Die Variablen sind über Adressen ansprechbar und können sowohl angezeigt als auch verändert werden. Der Debugger selbst ist ein C-Programm, das über eine spezielle Schnittstelle des Laufzeitsystems gerufen wird.

Vergleich zu anderen Compilern und zu BASIC

Im Vergleich zu schnellen C-Compilern unter CP/M bzw. SCP ist Pretty C etwa 2–3mal langsamer (bei ganzzahliger Arithmetik). Dafür ist der erzeugte Code oft etwas kürzer (beim KC 85/1 vielleicht der wichtigere Gesichtspunkt), und fast die volle Sprache wird schnell übersetzt, während andere auf 8-Bit-Rechnern verfügbare Compiler oft nur bis 16 Bit breite Datentypen verarbeiten oder sehr langsam übersetzen.

Pretty C sollte weniger als eine Nachentwicklung, sondern mehr als eine Alternative betrachtet werden. Im Vergleich zu BASIC sind die C-Gleitkommaprogramme etwa 3mal schneller, Programme mit ausschließlich ganzzahliger Arithmetik dagegen meist 30mal schneller. BASIC-Programme beanspruchen weniger Speicherplatz für den Code, aber mehr Arbeitsspeicher als C-Programme.

Weiterentwicklung und Vertrieb

Als Weiterentwicklung sind u. a. geplant:

– Initialisierungen in einfacher Form

– Auslagern übersetzter Programme zusammen mit dem Laufzeitsystem (5,5 K) auf Band

– Übertragung des Compilers auf CP/M- bzw. SCP-Rechner. Das ist für den Compiler selbst einfach, lediglich der Editor greift noch direkt zum Bildwiederholspeicher und Cursor zu.

Die Nachnutzung beinhaltet den Compiler mit Installierungsprogramm, 3 in C geschriebene Bibliotheken, den Debugger und eine etwa 110 Seiten lange Dokumentation.

Da bisher kein Betrieb gefunden werden konnte, der den Vertrieb der Software übernimmt, versucht der Autor auf dem Weg über diese Veröffentlichung, einen geeigneten Vertragspartner zu finden.

Literatur

/1/ Kernighan, B. W.; Ritchie, D. M.: Programmieren in C, Carl Hanser Verlag München/Wien 1983. (russ. Jazyk programirovan'a si, Finansy i Statistika, Moskau 1985)

KONTAKT

ZFW der AdW, Haeckelstr. 20, Dresden, 8027; Tel. 4 63 55 55

PASCAL

(Teil 4)

Dr. Claus Kofer
Informatikzentrum des Hochschulwesens
an der Technischen Universität Dresden

Die zur Darstellung insgesamt zu verwendenden Stellen werden durch m festgelegt, die nach dem Dezimalpunkt durch n. Beispiele mit den entsprechenden Ausgaben sind:

WRITE(OUTFILE,123:8) liefert:
123
WRITE(OUTFILE,1.5:8:3) liefert:
1.500
WRITE(OUTFILE,TRUE:8) liefert:
TRUE
WRITE(OUTFILE,'A':8) liefert:
A

6.5. Standardfiles

PASCAL stellt zwei vordeklarierte Files bereit:

VAR INPUT,OUTPUT: FILE OF CHAR;

Falls sie benutzt werden sollen, sind sie in der Environmentliste anzugeben.
Ihre Zuordnung zu Files im Sinne des Betriebssystems ist bei den PASCAL-Systemen unterschiedlich. Bei TURBO-PASCAL korrespondieren INPUT und OUTPUT mit dem Gerät CON:. Abweichend vom Standard brauchen sie nicht in der Environmentliste angegeben werden. Ebenso ist die Ausführung von RESET oder REWRITE verboten.
Die Arbeit mit dem Bildschirm wird bei TURBO-PASCAL durch die Standardprozedur GOTOXY(z,s) und CLRSCR zur Cursorpositionierung bzw. zum Bildschirmlöschen unterstützt.

6.6. Direktzugriffsfiles

Direktzugriffsfiles gibt es in Standard-PASCAL nicht. Viele Systeme stellen sie jedoch aufgrund ihrer Wichtigkeit bereit. Die gewählte Lösung ist sehr einfach: Die Komponenten werden fortlaufend numeriert. Der Zugriff auf sie erfolgt über die Komponentennummer. In verschiedenen Systemen wird die Komponentennummer n als zusätzlicher Parameter der Prozeduren GET und PUT angegeben:

GET(f,n) und **PUT**(f,n).

TURBO-PASCAL verwendet einen anderen Mechanismus: Vor dem Lesen oder Schreiben wird die Puffervariable mit der Standardprozedur

SEEK(f,n)

auf die Komponente n positioniert. Die Zählung der Komponenten beginnt bei Null.
Weiter gibt es in TURBO-PASCAL die Standardfunktionen zur Arbeit mit Direktzugriffsfiles, die folgende INTEGER-Resultate liefern:

FILEPOS(f) Aktuelle Position der Puffervariablen im File f.
FILESIZE(f) Gesamtanzahl der Komponenten des Files f.

7. Datentyp Array

7.1. Einführung

Der Datentyp Array wird durch eine feste Anzahl von Datenelementen gleichen Typs gebildet. Die einzelnen Datenelemente heißen Komponenten. Ihre Anzahl muß zur Übersetzungszeit feststehen.
Für den Typ der Komponenten gibt es keine Einschränkungen. Insbesondere kann auch er wieder ein Array sein.
Der Zugriff auf die Komponenten erfolgt mit Hilfe von Indizes. Indizes können berechnet werden.

7.2. Syntax

Die Arraydeklaration ist eine Alternative des Syntaxprogramms "typ". Sie wird im Bild 7.1 gezeigt.
Sie beginnt mit dem reservierten Bezeichner ARRAY. Dann folgt eine in eckige Klammern

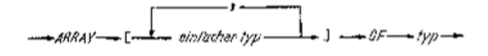


Bild 7.1 Syntaxdiagramm "arraytyp"

eingeschlossene Liste von einfachen Typen. REAL und INTEGER sind hier nicht zugelassen. Diese Liste gibt die Indizes an. Auf den reservierten Bezeichner OF folgt die Angabe des Komponententyps.
Die Syntax für den Zugriff zu den Komponenten eines Arrays zeigt die obere Alternative in Bild 5.13. Die Ausdrücke liefern den Wert der entsprechenden Indizes. Der Typ der Ausdrücke muß mit dem der Deklaration übereinstimmen.
Der nachfolgende Arraytyp besteht aus Komponenten vom Typ REAL. Der Indextyp ist Teilbereich des Basistyps INTEGER.

```
TYPE Index = 1...900;
     Vektor = ARRAY[Index]OF REAL;
```

Mit diesem Arraytyp können die Variablen

VAR X,Y,Z : Vektor;

vereinbart werden.
Zugriffe auf die Komponenten sind dann wie folgt möglich:

X[1] , X[3*I+K] , Y[I DIV 2], Z[SQR(I)−1]

Syntaktisch korrekt ist auch die Form

VAR X,Y,Z:ARRAY [1..100] OF REAL;

Hier ist jedoch der Typ der Variablen X,Y,Z anonym und kann zu keiner weiteren Deklaration genutzt werden.
Ein weiteres Beispiel für ein Array ist:

```
TYPE Tag= (Mo,Di,Mi,Don,Fr,Sa,So);
     Arbeitszeit=
     ARRAY [Tag] OF REAL;
VAR  MeiersZeit : Arbeitszeit;
```

Der Komponententyp ist REAL. Der Indextyp ist ein Aufzählungstyp. Zugriffe auf die Komponenten der Variablen MeiersZeit sind

MeiersZeit[Mo]
MeiersZeit[succ(Mo)]

Zur Deklaration von Arrays mit mehreren Indizes gibt es zwei Interpretationen des Syntaxdiagramms, s. Bild 7.1:

```
  ARRAY[index1] OF
    ARRAY [index2] OF..typ
und
  ARRAY[index1,index2,...] OF typ
```

Beide Formen sind korrekt und unterscheiden sich inhaltlich nicht.
Für den Zugriff zu den Komponenten mehrfach indizierter Arrays gibt es nach Syntaxdiagramm 5.13 ebenfalls zwei Formen, die sich inhaltlich nicht unterscheiden:

bezeichner [ausdruck1][ausdruck2]...
und
bezeichner [ausdruck1,ausdruck2,...].

Wesentlich ist, daß die Anzahl und der Typ der Ausdrucke mit denen bei der Deklaration übereinstimmen muß.
Eine quadratische Matrix läßt sich wie folgt deklarieren:

```
TYPE Index = 1..Max;
     Matrix = ARRAY [Index,Index]
                          OF REAL;
VAR  A : Matrix;
```

Der Zugriff zu den Komponenten hat dann die Form

A[1,3],A[I,J],A[3-I,I DIV 2]

Die Typen der Arrayindizes können natürlich auch alle voneinander verschieden sein:

```
TYPE Bsp =
     ARRAY[1..3,BOOLEAN,CHAR]
                   OF INTEGER;
VAR  X : Bsp;
```

Komponenten von X werden z. B. durch

X[2,TRUE,'A'], X[I,ODD(J),CHR(K)]. usw.

ausgewählt.
Die PASCAL-Systeme sichern, daß beim Zugriff auf eine Komponente die Indizes innerhalb ihrer Wertebereiche liegen. Das verlangsamt die Abarbeitung des Programms

und führt zu größerem Programmkode. Deshalb können diese Kontrollen meist ein- und ausgeschaltet werden. Bei TURBO-PASCAL erfolgt das durch die Compileroption ○R+ bzw. ○R−.
Standard-PASCAL sieht das Packen von Arrays vor. Der Compiler wird durch Einfügen der reservierten Bezeichnung PACKED vor ARRAY angewiesen, für die Komponenten des Array eine speicherplatzsparende interne Darstellung zu wählen. Das erfolgt z. B. dadurch, daß für Teilbereichstypen nur noch so viele Bits vorgesehen werden, wie zur Darstellung ihres Wertbereiches tatsächlich notwendig sind.
Der Zugriff auf gepackte Datenstrukturen ist deshalb langsamer.
Gepackte Arrays mit dem Komponententyp CHAR sind mit Zeichenketten verträglich.
TURBO-PASCAL führt kein Packen durch. Es überliest PACKED, aber Arrays mit dem Komponententyp CHAR sind mit Zeichenketten verträglich.

7.3. Operationen

PASCAL gestattet für Arrays gleichen Typs die Zuweisung und die Ausführung von Vergleichsoperationen.
Mit den Komponenten können die für ihren Typ zulässigen Operationen ausgeführt werden.
Das erste Beispiel zeigt die Zuweisung von Arrays:

```
TYPE Index = ...;
     Mat = ARRAY[Index,Index]
              OF REAL
VAR  A,B : Mat;
...
A := B;
...
```

Wichtig ist, daß die Variablen den gleichen Typ haben. Aus der Sicht des Compilers heißt das: Deklaration durch den gleichen Typbezeichner oder den gleichen impliziten Typ. Strukturelle Gleichheit wie z. B.

```
VAR A:ARRAY[Index,Index]
            OF REAL;
    B:ARRAY[Index,Index]
            OF REAL;
```

erkennt der Compiler nicht.
Vergleiche können häufig vorteilhaft auf Arrays mit dem Komponententyp CHAR angewendet werden. Das folgende Programmstück zeigt ein Beispiel, bei dem in einem Array von Namen ein Schlüssel gesucht wird. Namen und Schlüssel sind selbst Arrays:

```
TYPE Alpha = ARRAY[1..10]
                OF CHAR;
VAR  key:Alpha;
     Namen:ARRAY[1..100]
                OF Alpha;
     I:INTEGER;
...
I:=0;
REPEAT
I:=I+1;
UNTIL Namen[I] = Key;
...
```

Den Arrays mit dem Komponententyp CHAR können Zeichenketten zugewiesen werden. Die Länge der Zeichenkette muß dabei mit der Anzahl der Komponenten des Arrays übereinstimmen.

7.4. Programmbeispiel

Der Datentyp Array ist die programmiersprachliche Entsprechung der aus der Mathematik bekannten Vektoren und Matrizen. So bieten sich Programmbeispiele aus der linearen Algebra geradezu an. Trotzdem soll hier eine andere Anwendung gezeigt werden: In einem Textfile ist die Häufigkeit des Auftretens der Buchstaben A ... Z zu ermitteln. Die Häufigkeit dieser Zeichen wird in dem Array H mit folgender Deklaration gezählt:

```
H:ARRAY[CHAR]OF REAL
```

Das zeichenweise Lesen eines Textfiles ist bereits aus Abschnitt 6.5. bekannt. So erhält man mit wenigen Erweiterungen die im Bild 7.2 gezeigte programmtechnische Lösung.

```
PROGRAM Zaehl;
VAR C: CHAR;
    H: ARRAY[CHAR] OF REAL;
    F: TEXT;
BEGIN
  FOR C:='A' TO 'Z' DO H[C]:=0.0;
  ASSIGN(F,...); RESET(F);
  WHILE NOT EOF(F) DO BEGIN
   WHILE NOT EOLN(F) DO BEGIN
    READ(F,C); H[C]:=H[C]+1
   END;
   READLN(F)
  END;
  FOR C:='A' TO 'Z' DO
    WRITELN(C,H[C])
END.
```

Bild 7.2 Programm zum Zählen der Zeichen in einem Textfile

8. Datentyp Record

8.1. Einführung

Der Datentyp Record wird durch eine feste Anzahl von Datenelementen gebildet, deren Typ unterschiedlich sein kann. Die einzelnen Datenelemente eines Records heißen Felder. Zu ihnen kann über sogenannte Feldbezeichner zugegriffen werden. Feldbezeichner können nicht berechnet werden. Ihr Name ist fest im Programmtext verankert.
Für den Typ der Recordfelder gibt es keine Einschränkungen. Insbesondere können sie auch selbst wieder Records sein.

8.2. Syntax

Die Syntax des Recordtyps wird in den Bildern 8.1 bis 8.3 gezeigt. Die reservierten Bezeichner RECORD und END rahmen die Feldliste ein. Ihr Aufbau wird im Bild 8.2 gezeigt. Sie besteht aus einzelnen Feldern, die durch einen Feldbezeichner und eine Typenangabe gebildet werden. Falls aufeinanderfolgende Recordfelder den gleichen Typ haben, können ihre Bezeichner zu einer durch

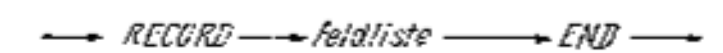


Bild 8.1 Syntaxdiagramm "recordtyp"

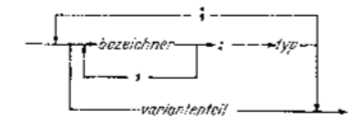


Bild 8.2 Syntaxdiagramm "feldliste"

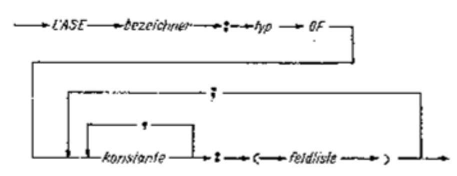


Bild 8.3 Syntaxdiagramm "variantenteil"

Komma getrennten Liste zusammengefaßt werden.
Die Feldliste kann durch einen Variantenteil abgeschlossen werden. Seine Behandlung erfolgt im Abschnitt 8.5.
Die Feldbezeichner haben nur innerhalb des Records Bedeutung. Sie überdecken bereits deklarierte Namen von Konstanten, Typen oder anderen Variablen nicht. Insbesondere können auch in verschiedenen Recordtypen gleiche Feldbezeichner verwendet werden.
Der Zugriff zu den Feldern eines Records ist bereits im Syntaxdiagramm in Bild 5.13 als untere Alternative dargestellt. Die einfachste Form sind zwei aufeinanderfolgende Bezeichner, die durch einen Punkt getrennt werden. Der erste Punkt muß der Name einer Recordvariablen, der zweite ein Feldbezeichner sein. Der Punkt hat die Aufgabe eines Selektors. Er wählt aus dem Record ein Feld aus. Ist dieses Feld selbst ein Record, kann ein weiterer Feldbezeichner angefügt werden. Das Syntaxdiagramm in Bild 5.13 erlaubt dies.
Es folgen einige Beispiele: Der Recordtyp

```
TYPE Complex = RECORD
       R : REAL;
       I : REAL
     END;
```

besteht aus zwei Feldern vom Typ REAL.
Die Feldbezeichner sind R und I. Da beide Felder den gleichen Typ haben, läßt sich auch abkürzend schreiben:

```
TYPE Complex = RECORD
       R,I : REAL
     END;
```

Vor dem abschließenden END ist lt. Syntax kein Semikolon erlaubt. Mit dem nachfolgenden Record wird ein Datentyp zur Beschreibung eines Datums eingeführt:

```
TYPE Datum = RECORD
       Jahr: 0..3000;
       Monat: (Jan,Feb,Mar,Apr,Mai,
       Jun,Jul,Aug,Sep,Okt,Nov,Dez);
       Tag: 1..31
     END
```

Der Record besteht aus drei Feldern mit den Bezeichnern Jahr, Monat und Tag. Die entsprechenden Datentypen dieser Felder sind Teilbereich von INTEGER, Aufzählungstyp und wieder Teilbereich von INTEGER.
Eine Komponente in einem File von Studenten könnte folgender Recordtyp Person sein:

```
Type  Alpha = ARRAY[1..12]
                    OF CHAR;
      Person = RECORD
        Name: Alpha;
        Vorname: Alpha;
        Semgr: 1..15
      END;
```

Zu ineinander geschachtelten Records kommt man sofort, wenn der Typ Person durch Angaben zum Datum der Geburt und der Immatrikulation erweitert wird.

```
TYPE  Alpha = ...;
      Datum = ...;
      Person = RECORD;
        Name: Alpha;
        Vorname: Alpha;
        GebDat: Datum;
        ImatDat: Datum;
        Semgr: 1..15
      END;
```

Der Zugriff auf die Felder der angegebenen Records wird am Beispiel der Variablen

```
VAR Y,Z: Complex;
    Jetzt: Datum;
    Student: Person;
```

gezeigt:
Y.R, Y.I, Z.R, Z.I
Jetzt.Jahr, Jetzt.Monat, Jetzt.Tag
Student.Name, Student.GebDat.Jahr,
Student.GebDat.Monat, usw.

8.3. Operationen mit Records

PASCAL gestattet für Records gleichen Typs die Zuweisung und die Vergleichsoperationen. Es wird noch einmal daran erinnert, daß die Compiler keine strukturelle Gleichheit erkennen. Siehe dazu auch Punkt 7.3.
Auf die Recordfelder können die für ihren Typ zulässigen Operationen angewendet werden.
Es folgen einige Beispiele auf der Grundlage der im vorigen Punkt eingeführten Recordtypen Complex, Datum und Person mit folgenden Variablen:

```
VAR  Y,Z:Complex;
     Jetzt: Datum;
     Student: Person;
     Jahrgang88:
       ARRAY[1..300]OF PERSON;
     ...
     Y:= Z; Y.R:=1.0;
     Y.R:=Y.R+Z.R; Y.I:=Y.I+Z.I;
```

Die letzte Zeile ist ganz offensichtlich die komplexe Addition, aber Y:=Y+Z; ist nicht erlaubt, da die Addition für Records nicht erklärt ist.

```
Jetzt.Jahr:= 1987;
Jetzt.Monat:= Jun;
Jetzt.Tag:= 11;
```

Das Datum, an dem diese Zeilen geschrieben wurden.

```
Student.Name:= 'Zimmermann';
Student.GebDat.Jahr:= 1968;
Student:= Jahrgang88[I]
Jahrgang88[I].Semgr:= 1;
```

Im letzten Beispiel wird zunächst eine Komponente im ARRAY Jahrgang88 ausgewählt. Sie ist ein Record, also kann mit dem Selektor ein Feld ausgewählt werden.

8.4. With-Anweisung

Die With-Anweisung gehört zu den Anweisungen, beeinflußt jedoch den Prozeßablauf nicht. Sie dient der Reduzierung von Schreibarbeit bei Bezugnahmen auf Recordfelder.
Das Bild 8.4 zeigt die Syntax der With-Anweisung. Ihre Wirkung besteht ausschließlich darin, daß sie für die zu ihr gehörende Anweisung die Feldbezeichnung der zwischen WITH und DO aufgeführten Recordvariablen bekannt macht.

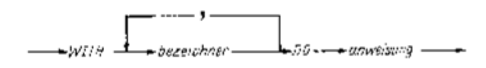


Bild 8.4 Syntaxdiagramm "with-anweisung"

Bei den gezeigten Beispielen darf unter Verwendung der With-Anweisung geschrieben werden:

```
WITH Y DO R:= 1.0
WITH Jetzt DO Jahr:= 1987
WITH Student DO
     Name:= 'Zimmermann'
```

Auch hier darf die Verbundanweisung benutzt werden, wenn eine einzige Anweisung nicht ausreicht:

```
WITH Jetzt DO BEGIN
  Jahr:= 1987;
  Monat:= Jun,
  Tag:= 11
END
```

Die allgemeine Form der Anweisung

WITH r1,r2,...rn DO anweisung

entspricht

```
WITH r1 DO
  WITH r2 DO
   ...
   WITH rn DO anweisung
```

Innerhalb der With-Anweisung können durch die Feldbezeichner eines Records andere Bezeichner überdeckt werden. Das folgende Programmstück zeigt dies:

```
TYPE Complex = ...;
VAR  I:BOOLEAN;
     X:Complex;
WITH X DO I:=1.0;
```

Der Feldbezeichner I überdeckt die Variable I vom Typ BOOLEAN. Auf sie kann innerhalb der With-Anweisung nicht Bezug genommen werden. Zur Beschleunigung des Zugriffs auf die Felder eines mit der With-Anweisung ausgewählten Records wird die Anfangsadresse des Records meist gesondert abgespeichert. Der dafür erforderliche Platz ist um so größer, je mehr With-Anweisungen verschachtelt sind. TURBO-PASCAL plant standardmäßig eine Tiefe von vier und reserviert im Datenbereich einer jeden Prozedur dafür Platz, unabhängig davon, ob diese Tiefe tatsächlich erreicht wird. Durch die OWN-Option kann der Standard auf eine Tiefe n zwischen 0 und 9 geändert werden.

8.5. Variantenrecords

Zur Motivation von Variantenrecords wird noch einmal der Record Person in seiner ersten Form aufgegriffen. Er enthält zur Kennzeichnung eines Studenten die drei Felder Name, Vorname und SemGr. Sollen mit dem Record Person auch Daten von anderen Beschäftigten erfaßt werden, hat das Feld SemGr keinen Sinn. Dafür werden aber andere Informationen benötigt, z. B. Wissenschaftsbereich, Gebäude, Zimmernummer und Telefon. Der Record Person sollte zwei verschiedene Bestandteile haben: einen feststehenden, mit den Feldern Name und Vorname, und einen, der entweder durch die Felder Seminargruppe oder Wohnheim gebildet wird, falls es sich um einen Studenten handelt oder durch Wissenschaftsbereich, Gebäude, Zimmernummer und Telefon, falls es ein Mitarbeiter ist. Eben diese Möglichkeit eröffnen Variantenrecords. Die Syntax zeigt Bild 8.3. Die Aufgabe der einzelnen Bestandteile wird am folgenden Beispiel gezeigt:

```
TYPE Alpha = ...;
     PersKat = (ST,MA);
     Heime = (WH1,WH2,WH3);
     StrEinh = (WB1,WB2,WB3,WB4);
     Bauwerk = (BW1,BW2);
     Person = RECORD
     Name, Vorname: Alpha;
     CASE PArt: PersKat OF
     ST:(SemGr: 1..15;
          Wohnh: Heime);
MA:       (WB:StrEinh;
           Geb: Bauwerk;
           Zi: 1..500;
           Tel: 100..999)
END;
```

Der reservierte Bezeichner CASE leitet die Varianten ein. Das Feld PArt ist das sogenannte Anzeigefeld. Es folgen nun die Varianten des Records. Dabei kann für jeden Wert des Anzeigefeldes eine Struktur angegeben werden. Im Beispiel sind das zwei.
Bei der Arbeit mit Variantenrecords ist der Programmierer für den korrekten Zugriff zu den Feldern des Variantenteils selbst verantwortlich. In der Regel wird dazu das Anzeigefeld zu Hilfe genommen.

Das folgende Programmstück zeigt dies:

```
VAR P: Person;
...
WITH P DO BEGIN
  IF PArt = St THEN
    WRITE ('SemGruppe=',SemGr)
  ELSE
    WRITE('Tel-Nr.=',Tel)
  END;
  ...
```

Falls die Unterscheidung der Varianten nicht notwendig ist, kann die Abspeicherung des Anzeigefeldes entfallen. Es wird bei der Deklaration nur sein Datentyp angegeben.
Variantenrecords lassen sich verwenden, um die Typenkontrollen des Compilers zu unterlaufen. Das folgende Beispiel zeigt den Zugriff zum nieder- und höherwertigen Byte eines Datenelementes vom Typ INTEGER:

```
TYPE Trick = RECORD
        CASE BOOLEAN OF
        FALSE:(Int:INTEGER);
        TRUE:(BY:ARRAY[0..1]
                          OF CHAR)
      END;
VAR I: Trick;
      High,Low:INTEGER;
...
I.INT:= ...;
Low:= ORD(I.BY[0]);
High:= ORD(I.By[1]);
...
```

Im Beispiel ist für das Anzeigefeld ein Typ notwendig, dessen Wertebereich aus zwei Werten besteht. Welcher das ist, hat keine Bedeutung. Hier wurde BOOLEAN gewählt, da er als Standardtyp bereits bekannt ist.

9. Datentyp Menge

9.1. Einführung

Eine Menge ist im mathematischen Sinne eine Zusammenfassung bestimmter, unterscheidbarer Elemente. Einschränkend wird in PASCAL gefordert, daß die Elemente alle vom selben einfachen Typ mit Ausnahme von REAL sind. So können z. B. die Mengen der geraden bzw. ungeraden ganzen Zahlen [2,4,6,8,10] bzw. [1,3,5,7,9] gebildet werden, nicht aber [1.5,TRUE,11], denn die Elemente gehören zu unterschiedlichen Datentypen. Die eckigen Klammern stellen hier den sogenannten Mengenkonstruktor dar. Die maximale Anzahl der Elemente einer Menge ist begrenzt. Zwischen den PASCAL-Systemen gibt es hier Unterschiede. Übliche Werte sind 64, 128 oder 256. TURBO-PASCAL beschränkt Mengen auf 256 Elemente. Den Wertebereich einer Variablen von Mengentyp bildet die Potenzmenge über der Basismenge. Die Potenzmenge enthält 2**n Teilmengen, wobei n die Anzahl der Elemente der Basismenge ist.

9.2. Syntax

Der Mengentyp ist eine Alternative von "typ" im Bild 3.1. Seine Syntax zeigt Bild 9.1. Die Deklaration von Mengen beginnt mit den reservierten Bezeichnern SET und OF. Es folgt die Angabe des Basistyps, der ein einfacher Typ (ohne REAL) sein muß.
Mengenkonstruktoren gestatten die Bildung von Mengenkonstanten. Sie können in Ausdrücken als Faktor verwendet werden. Ihre Syntax zeigt Bild 9.2. Es gibt zwei Grundformen:

[ausdruck, ausdruck, ...]
oder
[ausdruck .. ausdruck, ...]

In der ersten Form werden die Ausdrücke ausgewertet und die entsprechenden Werte des Basistyps in die Menge aufgenommen. Bei der zweiten Form werden die Ausdrücke ebenfalls ausgewertet. Die Resultate grenzen einen Bereich des Basistyps ein, aus dem alle Werte in die Menge aufgenommen werden. Es sind Kombinationen erlaubt, z. B.

[ausdruck, ausdruck .. ausdruck,ausdruck]

→ SET → OF → einfacher typ →

Bild 9.1 Syntaxdiagramm "mengentyp"

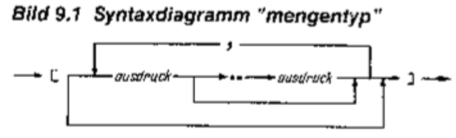


Bild 9.2 Syntaxdiagramm "mengenkonstruktor"

Es folgen Beispiele:

TYPE T=SET OF (A,B,C);

Der Wertebereich des Mengentyps T besteht aus

- den einelementigen Teilmengen **[A],[B],[C]**
- den zweielementigen Teilmengen **[A,B],[A,C],[B,C]**
- der Grundmenge **[A,B,C]**

und der Leeren Menge [ ]

Mehr als diese acht Werte (2**3=8) gibt es nicht. Mengenkonstruktoren für den Mengentyp T sind:

[A] , [A .. C] , [B,C] , [X]

Hier muß X eine Variable vom Typ T sein. Es wird eine Menge über einen Teilbereich von INTEGER gezeigt:

TYPE BitSet = **SET OF 0 .. 7;**

Der Wertebereich des Mengentyps BitSet besteht aus 256 Teilmengen (2**8=256). Erlaubte Mengenkonstruktoren sind:

[0]
[1]
[7]
[0,3,5 .. 7]
[0 .. 7]
[3*I+K]

Der Ausdruck 3*I+K muß einen Wert vom Typ INTEGER liefern und im Bereich 0 ... 7 liegen.
Mengen über dem Basistyp INTEGER können nicht deklariert werden, da der Wertebereich von INTEGER mehr als 256 Elemente umfaßt. Ein letztes Beispiel zeigt einen Mengentyp über den Basistyp CHAR:

TYPE CharSet = **SET OF CHAR;**

Wenn von 127 Zeichen des ASCII-Kodes ausgegangen wird, besteht der Wertbereich des Datentyps CharSet aus mehr als 1.R38 (genau 2**127) Elementen. Mengenkonstruktoren vom Typ CharSet sind z. B.

['A' .. 'Z']
['0' .. '9']
['+','-','*','/']
['a' .. 'z','+','/']
[C]
['z' .. C]

Die Variable C muß vom Typ CHAR sein.

9.3. Operationen mit Mengen

Für Datenobjekte vom Mengentyp gibt es folgende arithmetische Operationen:

\+ Vereinigung
* Durchschnitt
\- Differenz.

Weiterhin sind alle Vergleichsoperatoren anwendbar. Dabei wird getestet auf

< echte Teilmenge
<= Teilmenge
=,<> Gleichheit, Ungleichheit
>= Obermenge
> echte Obermenge.

Durch den Operator IN kann weiterhin abgefragt werden, ob ein bestimmtes Element in einer Menge enthalten ist.
Falls M1, M2 und M3 Mengen des Typs SET OF (A,B,C) sind und die Werte M1=[A,B],M2=[B,C] und M3=[C] haben, liefern die Verknüpfungen folgende Ergebnisse:

```
M1 + M2   = [A,B,C]
M2 + M3   = [B,C]
M1 * M2   = [B]
M1 * M3   = [ ]
M2 - M3   = [B]
M1 - M2   = [A]
M3 < M1   = TRUE
M1 <=M1   = TRUE
M1 = M2   = FALSE
M1 <> M2  = TRUE
M2 >=M2   = TRUE
M2 > M3   = TRUE
A IN M1   = TRUE
C IN M1   = FALSE
```

wird fortgesetzt

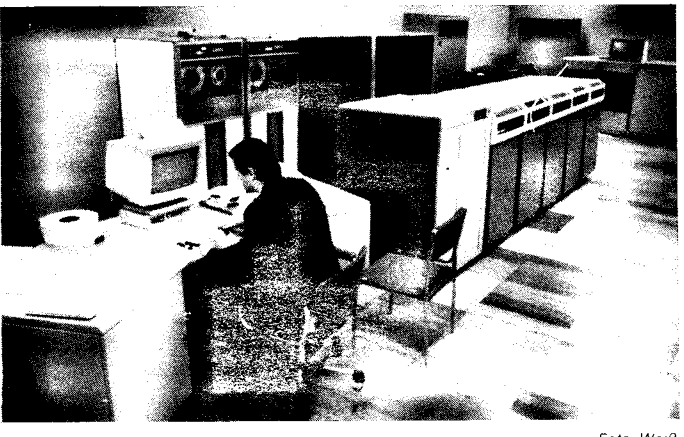


Foto: Weiß

ISOT 1014E (EC 1037)

Am 12. Oktober 1987 berichtete die amerikanische Zeitschrift „Aviation Week and Space Technology" wie folgt über den Einsatz eines „Supercomputers" in der UdSSR: „Ein großer sowjetischer Computer, der mit 10 parallelen Prozessoren, angeschlossen an einer IBM-kompatiblen zentralen EDV-Anlage, arbeitet, wurde im Institut für Weltraumforschung bei der Akademie der Wissenschaften der UdSSR in Betrieb genommen. Das ist das erste System der Familie Super-Computer in der UdSSR.

Das System, das vor etwa 6 Monaten in Betrieb genommen wurde, hat eine theoretische maximale Leistung von etwa 120 Mio normalen Operationen je Sekunde.

Das wird es dem Institut erlauben, komplizierte theoretische und angewandte Probleme zu lösen sowie zu modellieren. Das System wurde bereits zur Lösung solcher komplizierten Probleme wie der Erforschung von turbulenten Strömungen benutzt. Die Computeranlage wurde von der bulgarischen Vereinigung ISOT geliefert, die Software im Institut für Weltraumforschungen erarbeitet." Während der Leipziger Frühjahrsmesse nun war ein solcher Rechner auf dem bulgarischen Ausstellungsstand zu besichtigen (siehe Bild), so daß wir hier ausführlicher berichten können.

Der Rechner ISOT 1014 E (EC 1037) wird seit 1986 in der VR Bulgarien hergestellt; die konstruktive und technologische Ausführung des Systems widerspiegelt die neuesten Erkenntnisse auf dem Gebiet der Computertechnik und entspricht dem Weltstand bei Computern mittlerer Leistung (ESER Reihe 3).

Kern des Systems ist die zentrale Verarbeitungseinheit ISOT 2100E. Die ZE besitzt folgende Merkmale:

– Operationsgeschwindigkeit 1 Mio Op/s
– Hauptspeicherkapazität bis 16 MByte aufrüstbar
– 6 Kanäle in der Basiskonfiguration (1 BYMPX, 5 BLMPX)
– sehr niedrige Anschlußleistung von 3,5 kW
– Betriebssystem VM/35.

Die gesamte Einheit, einschließlich der Konsole mit dem Bildschirmgerät, Hardcopy-Drucker und der Stromversorgung, ist als eine Einschrank-Konstruktion ausgeführt, was die Bedienung und Aufstellung der Anlage erleichtert. Die Verwendung moderner Bauelemente hat es ermöglicht, die Basiskonfiguration der ZE auf nur 14 Steckeinheiten zu realisieren. Der Hauptspeicher (16 MByte) ist mit 256-KBit-RAMs realisiert worden.

Jeder der Kanäle befindet sich auf einer separaten STE und kann sowohl bytemultiplex (BYMPX) als auch blockmultiplex (BLMPX) arbeiten. Durch Erweiterungskarten kann die Zahl der Kanäle auf 12 erhöht werden. Im BLMPX-Regime hat der Kanal eine Übertragungsgeschwindigkeit von 2 MByte/s.

Die hohe Leistung der ZE ISOT 2100E erfordert natürlich Hochleistungsperipheriegeräte. Für die ISOT 1014E wurde eine Reihe solcher Geräte entwickelt. Der Matrix-Prozessor EC 2706 zum Beispiel arbeitet nach der Fließband-Organisation und ist für Aufgaben, die hohen Datendurchsatz verlangen, besonders geeignet.

Der EC 2706 besitzt einen eigenen Operativspeicher, der bis auf 16 MByte ausgebaut werden kann. Bei Aufgaben wie z. B. schnelle Fourier-Transformation arbeitet der EC 2706 etwa 500mal schneller als der Rechner EC 1035, d. h. etwa mit 100 Mio Op/s. Besonders wichtige Teile der ISOT 1014 E sind das 317-MByte-Plattenspeichersubsystem und das Magnetbandsystem EC 5527/5027, die dem modernsten Stand der Speichertechnik entsprechen.

Das 317-MByte-Subsystem besteht aus dem neuentwickelten Steuergerät EC 5563, dem Steuermodul EC 5663 und dem Festplattenspeicher EC 5063. In Vorbereitung sind die 635-MByte-Festplattenspeicher EC 5065, aufbauend auf der Konstruktion des EC 5063. Die Zylinderzahl wird hierbei von 561 auf 1121 erhöht, die Spurdichte beträgt 378 Spuren/cm.

Als Betriebssystem für die ISOT 1014E wurde das VM/35 entwickelt. Es ist ein System virtueller Maschinen, das jedem Anwender eine eigene EDV-Anlage mit der Architektur der ESER-Reihe 1 bzw. Reihe 2 zur Verfügung stellt. Ein integrierter Teil des VM/35 ist das Dialogsystem – CMS, das dem Anwender günstige Bedienungsmöglichkeiten bietet.

Mit den Komponenten der ISOT 1014E werden Mehrrechnerkomplexe aufgebaut, die einen besonders hohen Datendurchsatz erreichen. An einem solchen Komplex mit zwei Zentraleinheiten ISOT 2100 E und insgesamt 32 MByte Hauptspeicher, 8 Matrixprozessoren EC 2706 und 18 GByte Magnetplattenspeicherkapazität beträgt die Operationsgeschwindigkeit 96 Mio Gleitkomma-Op/s. Bei Matrix- und Vektorenoperationen kann eine Operationsgeschwindigkeit von 320 Mio Op/s erreicht werden. Dabei beträgt die Übertragungsgeschwindigkeit zwischen den Matrixprozessoren 12 MByte/s.

Als Programmiersprachen können PAL (Makroassembler) und FORTRAN 77 genutzt werden.

MP

TERMINE

Fachtagung „Computer- und Mikroprozessortechnik '88"

WER? Fachverband Elektrotechnik in der Kammer der Technik, Wissenschaftliche Sektion Computer- und Mikroprozessortechnik
WANN? 15. – 16. Dezember 1988
WO? Berlin
WAS?
– Mikroprozessorsysteme/Kundenschaltkreise
– Entwurfs- und Testsysteme
– Expertensysteme
– PC (16 Bit), Mikrorechnersysteme (32 Bit)
– Mikroprozeßrechner-/Steuerungssysteme
– Betriebssysteme/Fachsprachen
WIE? Vorträge bis zum 28. Juni 1988 sowie Teilnahmemeldungen sind schriftlich an folgende Anschrift zu richten: Kammer der Technik, Präsidium, Fachverband Elektrotechnik, Clara-Zetkin-Str. 115/117, Berlin, 1086, Tel.: 2 26 52 85

Hoppe

Sequentielle Online-Verarbeitung von dBASE-II-Dateien mit TURBO-PASCAL

Christian Hanisch
Technische Universität Dresden, Sektion Wasserwesen

Für technisch-wissenschaftliche Anwendungen ist die Kombination aus dBASE II zur standardisierten Datenerfassung und -verwaltung und eigenprogrammierten Auswerteprogrammen in TURBO-PASCAL dann von Vorteil, wenn die dBASE-Datenbankdatei direkt von TURBO aus gelesen werden kann (Online-Verarbeitung).

Nachfolgend soll als Beispiel ein leicht abwandelbares Muster für eine ONLINE-Verarbeitung von dBASE-Dateien gezeigt werden. Neben dem Aufbau der dBASE-Metadaten im Struktursatz zeigt das Beispiel die Technik der Arbeit mit ungetypten Files (BLOCKREAD), mit dynamischer Speicherverwaltung und Pointervariablen. Wegen der Austauschbarkeit der Prozedur **VERARBEITE** (Bild 3) sowie der Datenstruktur **Datadat** (Bild 2) stellt das Programm **XDBASE.PAS** (Bild 1) ein universell anpaßbares Rahmenprogramm dar.

Um den Entwicklungsprozeß der Formulierung eines Programms zu demonstrieren, ist als Test-Variante die Verarbeitung einer Datenbank **MAWI.DBF** zugrundegelegt, deren Struktur und problemspezifische Verarbeitung in Bild 4 (Screen-Dump) gezeigt ist.

Bilder siehe Seiten 180 und 181

✉ KONTAKT ☏

Technische Universität Dresden, Sektion Wasserwesen, Bereich WE, Mommsenstraße 13, Dresden, 8027; Tel. 2 32 61 18

```
  1:  0 Program XDBASE;
  2:  0 {***********************************************************}
  3:  0 { Einlesen und Verarbeiten von dBASE-Datenbankdateien:      }
  4:  0 { Satzlaenge, Namen der Felder sowie Feldtyp und Anzahl     }
  5:  0 { Saetze in der Datei und das Datum der letzten Aenderung   }
  6:  0 { an den Daten aus dem Struktursatz entnehmbar.             }
  7:  0 { (Metadat und FeldDat ---> 8 Byte + 32*16 Byte = 520 Byte)}
  8:  0 {***********************************************************}
  9:  0 {$U+ }
 10:  0 LABEL 1;
 11:  0 CONST Max:INTEGER=4096;  {Vielfache von 128 Byte (Sektoren)}
 12:  0 TYPE
 13:  0
 14:  0 { ============== dBASE-Struktursatz ===== BEGIN ======== }
 15:  0 FeldDat=RECORD
 16:  1          Fname     : ARRAY[1..10] OF CHAR; {Ende --> 0Dh
 17:  1                                      im 1. Byte von Fname }
 18:  1          Fill1     : BYTE;          { Binaer Null          }
 19:  1          Ftype     : CHAR;          { Typ: C ¦ N ¦ L       }
 20:  1          Flength   : BYTE;          { Feldlaenge insgesamt }
 21:  1          Fill2     : INTEGER;
 22:  1          Fdec      : BYTE           { Anzahl Dezimalstellen }
 23:  1        END;
 24:  0
 25:  0 Metadat=RECORD   { 8 Byte + 32 * 16 Byte        }
 26:  1          Version : BYTE;          { Versions-Nr. von dBASE }
 27:  1          Count   : INTEGER;  { Anzahl Saetze in der Datei }
 28:  1          Datumx  : ARRAY[1..3] OF BYTE; { Binaer tt mm jj }
 29:  1          Recsize : INTEGER;                  { Satzlaenge }
 30:  1          FieldId : ARRAY[1..32] OF FeldDat;
 31:  1          Fill3   : BYTE;        { 00h --> Ende Struktursatz }
 32:  1          EOD     : BYTE;   { Erstes Datenbyte (LKennz); bei
 33:  1                               leerer Datenbasis LKennz:=$1A }
 34:  1          Fill4   : ARRAY[1..118] OF BYTE
 35:  1              { Auffuellen auf 5*128 Byte = 521 + 119 Byte }
 36:  1        END;
 37:  0
 38:  0 { ==== dBASE-Struktursatz ( 521 Byte am Dateibeginn ) ==== }
 39:  0
 40:  0 DPB    = ^BYTE;
 41:  0 BufPtr = ^Buf;
 42:  0 Buf    = ARRAY[1..1] OF BYTE;     { Dynamischer Puffer
 43:  0                                   mit GETMEM angefordert }
 44:  0 DataPtr= ^Datadat;
 45:  0
 46:  0 {$I DATADAT.INC <-- Strukturbeschreibung der USEd Datenbank}
 47:  0
 48:  0 VAR  dsname            : STRING[14];
 49:  0      DBF_file          : file;   {dBASE-Datei als Untyped File}
 50:  0      Zaehler,LRECL,
 51:  0      PA,PE,PR,I,J,
 52:  0      K,L,M,N,NX,Rest:  INTEGER;
 53:  0      Struktur_Satz     : Metadat;
 54:  0      Datensatz         : Datadat;
 55:  0      PXP               : BufPtr;{Puffer-Pointer fuer BLOCKREAD}
 56:  0      GroupKey          : DPB;
 57:  0      RecBuf            : Kette;
 58:  0      PtrX,PtrZ         : DataPtr;
 59:  0      PtrY              : ^Kette;
 60:  0      X                 : CHAR;
 61:  0      EOFflag           : BOOLEAN;
 62:  0
 63:  0 PROCEDURE LIESNBLOCKS;
 64:  0 {  Einlesen von N*128 Byte in den dynamischen Puffer PXP^  }
 65:  0 VAR NX: INTEGER;
 66:  0 BEGIN
 67:  1   nx:=N;
 68:  1   IF NX<=Rest THEN BLOCKREAD(DBF_file,PXP^,NX)
 69:  1              ELSE BEGIN NX:=Rest;
 70:  2                   BLOCKREAD(DBF_file,PXP^,NX) END;
 71:  1   Rest:=Rest-NX;
 72:  1   IF Zaehler>0 THEN FOR I:=1 TO M DO RecBuf[I+L]:=PXP^[I];
 73:  1   PE:=ADDR(RecBuf);PtrZ:=PTR(PE);PA:=ORD(PXP);
 74:  1   PE:=PA+NX*128;            { Endeadresse+1 des Puffers    }
 75:  1   PA:=PA+M;
 76:  1   PtrX:=PTR(PA);
 77:  1   K:=(PE-PA) DIV LRECL;     { Anzahl ganze Saetze im Puffer }
 78:  1   L:=(PE-PA) MOD LRECL;     { Laenge  Restsatz im Puffer    }
 79:  1   PR:=PA+K*LRECL;
 80:  1   PtrY:=PTR(PR);             {Startadresse Satzrest im Puffer }
 81:  1 END;
 82:  0
 83:  0 {$I VERARB.INC  <-- VERARBEITUNG der USEd Datenbank        }
 84:  0
 85:  0 { ==================== Hauptprogramm ================== }
 86:  0 BEGIN
 87:  1   IF MEM[$80]=$FF THEN GOTO 1;             { dsname von CHAIN }
 88:  1   WRITE(#12'ENTER [<d>:]<filename_DBF>[.<type>] : ');
 89:  1   READLN(dsname);
 90:  1 1:IF dsname='' THEN dsname:='MAWI.DBF';     { Testdatenbank }
 91:  1   IF POS('.',dsname)=0 THEN dsname:=dsname+'.DBF';
 92:  1   ASSIGN (DBF_file,dsname);
 93:  1   RESET(DBF_file);
 94:  1   Rest:=FILESIZE(DBF_file);
 95:  1   BLOCKREAD(DBF_file,Struktur_Satz,5);
 96:  1   Rest :=Rest-4;
 97:  1   LRECL:=Struktur_Satz.Recsize;
 98:  1   FOR J:=1 TO 32 DO FOR I:=1 TO 10 DO
 99:  1     IF Struktur_Satz.FieldId[J].Fname[I]=^@  THEN
100:  1                      Struktur_Satz.FieldId[J].Fname[I]:=' ';
101:  1   SEEK(DBF_file,4); { Auf "ersten" Datensatz-Sektor stellen }
102:  1   WRITELN; WRITELN(#134'Strukturdaten: '#132); WRITELN;I:=1;
103:  1   WHILE (Struktur_satz.FieldId[I].Fname[1]<>#$0D) AND (I<=32)
104:  1   DO BEGIN
105:  2     WRITE('Feld-Nr.: ',I:2,' ¦ ');
106:  2     FOR J:=1 TO 10 DO WRITE(Struktur_Satz.FieldId[I].Fname[J]);
107:  2     WITH Struktur_Satz.FieldId[I] DO
108:  2     WRITE(',',Ftype,',',Flength);
109:  2     IF Struktur_Satz.FieldId[I].Ftype='N' THEN
110:  2                 WRITE (',',Struktur_Satz.FieldId[I].Fdec);
111:  2     WRITELN; I:=I+1
112:  2   END;
113:  1   WRITELN;
114:  1   WRITELN('Anzahl Saetze in der Datei: ',Struktur_Satz.Count);
115:  1   WRITELN('Satzlaenge: ',Struktur_Satz.Recsize);
116:  1   WRITELN('Datum der letzten Aenderung: ',
117:  1     Struktur_Satz.Datumx[1]:2,'.',Struktur_Satz.Datumx[2]:2,
118:  1     '.19',Struktur_Satz.Datumx[3]:2);
119:  1   FOR I:=1 TO LENGTH(dsname) DO dsname[I]:=UPCASE(dsname[I]);
120:  1   IF Struktur_Satz.EOD=$1A THEN BEGIN WRITELN;
121:  2     WRITELN('Datenbasis '#134,dsname,#132'leer --> CANCEL');
122:  2     HALT END;
123:  1   READLN(X);            { Zwecks Pausieren --> Weiter mit <CR> }
124:  1   {     Zugriffsoptimierung: --- > Optimale Transferrate     }
125:  1   IF Max<>2048 THEN BEGIN
126:  2     WRITELN; WRITE('ENTER N fuer N*128-Transferrate: ');
127:  2     READLN(N) END
128:  1   ELSE BEGIN
129:  2     I:=BDOSHL(31)+2; GroupKey:=PTR(I); N:=Max DIV 128;
130:  2     IF GroupKey^=3 {1 K Aufzeichnungsbloecke} THEN N:= 8;
131:  2     IF GroupKey^=4 {2 K Aufzeichnungsbloecke} THEN N:=16;
132:  2   END;
133:  1   IF LRECL>N*128 THEN N:=LRECL DIV 128 + 1;
134:  1   { IF (ORD(MEMAVAIL<0)*65536+MEMAVAIL>N*128) THEN  }
135:  1                          GETMEM(PXP,N*128);
136:  1   { READ ahead des fuenften 128-Byte-Blockes (N*128-Block) }
137:  1   Zaehler:=0; M:=9; EOFflag:=false; LIESNBLOCKS;
138:  1
139:  1 { ==================== Hauptzyklus ==================== }
140:  1   REPEAT
141:  2     IF (Zaehler>0) AND (M>0) AND (M<LRECL) AND
142:  2       (PtrZ^.LKennz<>#$1A) THEN BEGIN
143:  3       Zaehler:=Zaehler+1; VERARBEITE(PtrZ,'!'); READ(X) END;
144:  2     FOR I:=1 TO K DO BEGIN
145:  3       IF PtrX^.LKennz<>#$1A {DateiENDe ::= LKennz=1Ah} THEN
146:  3       BEGIN ZAEHLER:=Zaehler+1; VERARBEITE (PtrX,'¦');
147:  4         READ(X);                  { Pausieren nach jedem Satz }
148:  4         PA:=PA+LRECL; PtrX:=PTR(PA)
149:  4       END
150:  3       ELSE BEGIN
151:  4         IF NOT(EOFflag) THEN WRITELN('**** Dateiende ****');
152:  4         EOFflag:=true END
153:  3     END;
154:  2     FOR I:=1 TO L DO RecBuf[I]:=PtrY^[I];
155:  2     M:= LRECL - L; LIESNBLOCKS;
156:  2   UNTIL EOFflag;
157:  1   CLOSE(DBF_file); FREEMEM(PXP,N*128); READ(X)
158:  1 END.
```

1

```
{DATADAT.INC:      Statt STRING[5] --> ARRAY[1..5] OF CHAR;
 ===========}

{ ============= Datenbank spezifisch ======= BEGIN ======= }
Datadat= RECORD
          LKennz: CHAR;{Stets vorhanden 20h ¦ 2Ah
                            oder 1Ah fuer Dateiende        }
          ANR: ARRAY[1..12] OF CHAR;
          BEZ: ARRAY[1..10] OF CHAR;
          STK: ARRAY[1..5] OF CHAR;
          ME:  ARRAY[1..2] OF CHAR;
          PR:  ARRAY[1..5] OF CHAR;
          SUZ: ARRAY[1..5] OF CHAR;
          SUA: ARRAY[1..5] OF CHAR;
          BEST:ARRAY[1..5] OF CHAR
        END;
Kette=ARRAY[1..50] OF BYTE;     { Laenge LRECL aus Recsize }
      { Die Struktur Datadat als Zeichenkette }
{ ============= Datenbank spezifisch ======= END ========= }
```

2

```
{VERARB.INC:
 ==========}

PROCEDURE VERARBEITE (Pointer: DataPtr;T: CHAR);
VAR Wert,Preis,Bestand: REAL;  C1,C2:INTEGER;
BEGIN                   { Verarbeitung: Datenbank spezifisch }
 FOR C1:=1 TO 5 DO BEGIN
   IF Pointer^.PR[C1]=' '   THEN Pointer^.PR[C1]:='0';
   IF Pointer^.BEST[C1]=' ' THEN Pointer^.BEST[C1]:='0' END;
 VAL(CONCAT(Pointer^.PR[1],Pointer^.PR[2],Pointer^.PR[3],
            Pointer^.PR[4],Pointer^.PR[5]),Preis,C1);
 VAL(CONCAT(Pointer^.BEST[1],Pointer^.BEST[2],Pointer^.BEST[3],
            Pointer^.BEST[4],Pointer^.BEST[5]),Bestand,C2);
 Wert := Preis * Bestand;
 WRITE(Pointer^.ANR,' ',T,' ',Pointer^.BEZ,' ',T,' ',
       Pointer^.BEST:5,'  ** '#134,Wert:9:2,' **'#132);
 WRITELN ('  : ',Zaehler:4,'.');
END;
```

3

Bild 1 Rahmenprogramm XDBASE.PAS

Bild 2 Der datenbankspezifische INCLUDE-Bestandteil

Bild 3 Der problemspezifische INCLUDE-Bestandteil

Bild 4 Anzeige der Strukturdaten der dBASE-Datei MAWI.DBF

```
ENTER [<d>:]<filename_DBF>[.<type>] : c:mawi.dbf

Strukturdaten:

Feld-Nr.:  1 ¦ ANR       ,C,12
Feld-Nr.:  2 ¦ BEZ       ,C,10
Feld-Nr.:  3 ¦ STK       ,N,5,1
Feld-Nr.:  4 ¦ ME        ,C,2
Feld-Nr.:  5 ¦ PR        ,N,5,2
Feld-Nr.:  6 ¦ SUZ       ,N,5,1
Feld-Nr.:  7 ¦ SUA       ,N,5,1
Feld-Nr.:  8 ¦ BEST      ,N,5,1

Anzahl Saetze in der Datei: 23
Satzlaenge: 50
Datum der letzten Aenderung: 18. 3.1987
* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
ENTER N fuer N*128-Transferrate: 1
5555-333-111 ¦ Hammer     ¦ 004.0   **      6.00 **  :    1.
7777-999-000 ¦ Schraube   ¦ 006.0   **      3.00 **  :    2.
1111-333-666 ! Zange      ! 006.0   **      9.00 **  :    3.
9999-888-222 ¦ Mutter     ¦ 002.0   **      0.10 **  :    4.
4444-555-111 ! Schlauch   ! 009.0   **     13.50 **  :    5.
2222-444-000 ¦ Scheibe    ¦ 011.0   **      2.75 **  :    6.
6666-777-222 ¦ Nippel     ¦ 011.0   **      4.95 **  :    7.
                ...                 ...
6666-111-111 ! Reifen     ! 020.0   **    468.00 **  :   23.
**** Dateiende ****
```

4

Änderungen am Betriebssystem SCP 1700 des AC A 7100

Michael Herse
VEB IFA Getriebewerke Brandenburg
Frank Isekeit
Deutsche Reichsbahn, Ingenieurbüro für Rationalisierung des Eisenbahnbaues

Es werden einige Änderungen am Betriebssystem SCP 1700 beschrieben, die die Gebrauchseigenschaften des A 7100 verbessern sollen.
Insbesondere werden das Einfügen bzw. Ändern von Assemblerroutinen im BIOS-Quelltext für
– Abarbeitung eines Kaltstartkommandos
– Einrichten und Initialisieren der RAM-Disk ohne Bedienerhandlungen
– Arbeiten mit 780 K-Disketten ab Kaltstart dargestellt.

Im Vergleich zu den gebräuchlichen CP/M-kompatiblen Betriebssystemen für den Personalcomputer PC 1715 fehlen dem Betriebssystem SCP 1700 einige vorteilhafte bzw. nutzerfreundliche Eigenschaften. Dazu zählt insbesondere die Möglichkeit der Nutzung eines sogenannten Kaltstartkommandos, d. h. des Abarbeitens eines Programmes oder Kommandos ohne Bedienereingriff nach dem Einschalten oder nach RESET. Probleme treten nach den Erfahrungen der Autoren auch bei der Nutzung der RAM-Diskette durch Bediener ohne EDV-Kenntnisse auf. In Abhängigkeit vom Einsatzzweck ist hier eine Vereinfachung der Bedienung wünschenswert.

Im Interesse der optimalen Ausnutzung der Diskettenkapazität und der Kompatibilität zum SCP 0.5 erscheint es sinnvoll, das Diskettenformat 780 KByte bereits beim Systemstart einzustellen und grundsätzlich in diesem Format zu arbeiten. Da ein Installationsprogramm zur einfachen Anpassung an die Forderungen des Nutzers (noch?) nicht zur Verfügung steht, wurden die angesprochenen Eigenschaften durch Änderungen im mitgelieferten BIOS-Quelltext beeinflußt.

Kaltstartkommando

Das Prinzip der vorgeschlagenen Variante besteht darin, nach Abschluß der Systeminitialisierung und noch vor Übergabe der Steuerung an den CCP mit Hilfe einer Assemblerroutine ein Kaltstartkommando in den Kommandopuffer des CCP einzutragen. Bild 1 zeigt die entsprechende Programmstrecke und verdeutlicht gleichzeitig, an welcher Stelle sie in den Quelltext einzufügen ist (Hinweise: die Marke SREND befindet sich fast am Ende des BIOS!). Alle benötigten Register werden gerettet, der Transport des Kaltstartkommandos erfolgt byteweise und wird abgebrochen, sobald ein NUL-Byte in der Zeichenkette auftritt. Es ist zu beachten, daß die Segmentadressen (1040H) des Kommandopuffers und des Speicherplatzes zum Speichern der Befehlslänge aus der Initialisierungsroutine des BIOS übernommen werden. Das für den Kaltstart ausgewählte Kommando wird im Datenspeicher-

Dipl.-Ing. Frank Isekeit (27) studierte von 1982 bis 1987 an der TU Dresden, Sektion Informationstechnik. Seit 1987 arbeitet er als Themenbearbeiter im Ingenieurbüro für Rationalisierung des Eisenbahnbaus.

Dipl.-Ing. Michael Herse (27) studierte von 1982 bis 1987 an der TU Dresden, Sektion Informationstechnik. Seit 1987 arbeitet er als Projektant für dezentrale Rechentechnik im VEB IFA Getriebewerke Brandenburg.

```
;****************          Original BIOS             ****************

SREND:  POP     CX                          ;GET DEVICE NUMBER
        MOV     BOOTDV,CL                   ;REMEMBER BOOT-DEVICE

        IF NOT LOADER_BIOS
;------------------------------------------------------------------
        MOV     BX,CCPOFFSET+BODVOFST       ;STORE BOOT-DEV.IN BDOS
        MOV     [BX],CL

;****************      EINFUEGEN KALTSTARTKOMMANDO       ****************

        PUSH    ES                  ;Retten der Register
        PUSH    DI                  ;Vorbeugung fuer andere Aenderungen
        PUSH    SI
        PUSH    AX
        PUSH    CX
        PUSHF
        MOV     AX,CS
        MOV     ES,AX               ;Segmentadresse einstellen
        MOV     SI,OFFSET KALTST    ;Offset Quelladresse Kommandotext
        MOV     DI,CCPBUFF          ;Speicherpl. f. Befehlslaenge
        PUSH    DI                  ;retten
        INC     DI                  ;Anfangsadresse Puffer
        CLD
        XOR     CX,CX
MARKE:  LODSB
        CMP     AL,Ø                ;Ende?
        JZ      MARKE1
        STOSB
        INC     CL
        JMPS    MARKE
MARKE1: POP     DI
        MOV     [DI],CL             ;Befehlslaenge speichern
        POPF
        POP     CX
        POP     AX
        POP     SI
        POP     DI
        POP     ES
;Ende der eingefuegten Routine
;------------------------------------------------------------------
        ENDIF

        JMP     CCP
```

Bild 1 Assemblerroutine zum Einfügen eines Kaltstartkommandos

Bild 2 Einstellung der Betriebsmodi des Bildschirmes, Systemausschriften (Auszug aus Datei BIOS.A86 mit Änderungen)

```
        IF      NOT     LOADER_BIOS
;------------------------------------------------------------------
SIGNON          DB      CR,LF,CR,LF
                DB      1BH,'[?7h'          ;Betriebsmodi des
                                            ;Bildschirmes
                DB      'SYSTEM GENERATED 2Ø-MAR-87'
                DB      CR,LF,'MEMORY SIZE :',Ø
SIGNON1         DB      '128',Ø
                DB      '256',Ø
                DB      '384',Ø
                DB      '512',Ø
                DB      '64Ø',Ø
                DB      '768',Ø
                DB      '896',Ø
SIGNON2         DB      ' K BYTES',CR,LF,Ø
SIGNON3         DB      CR,LF,'INSERT 78ØK- DISK IN BOOTDEVICE,
                DB      ' TYPE <CR> TO CONTINUE',Ø
ELDSKØ          DB      'DO YOU WANT A MEMORY-DISK E: LIKE K56ØØ.2Ø'
                DB      ' (3Ø6 K BYTES FOR USER) ? (Y/<CR>):',Ø
ELDSK1          DB      CR,LF,'INITIALIZE MEMORY-DISK ? (Y/<CR>):',Ø
;ELDSK1 aendern bei automatischem Einrichten der RAM- Diskette!
BOOTER          DB      CR,LF,'Press RESET-key to BOOT SCP 17ØØ correctly !'
                DB      CR,LF,'Otherwise BREAK/M-function is impossible.',Ø
KGSID           DB      1BH,'[Øc',Ø
ZKINDA          DB      3,3,7,1             ;TIME CHANNELS
S2INDA          DB      1Ø,Ø,18H,1,Ø,3,41H,4,45H,5,28H   ;V.24
IFINDA          DB      1Ø,Ø,18H,1,Ø,3,41H,4,45H,5,28H   ;IFSS
KALTST          DB      'SUBMIT START',Ø
;------------------------------------------------------------------
```

```
/~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\
ENTER N fuer N*128-Transferrate: 1                                  |
5555-333-111 | Hammer      | 004.0  **      6.00 **  :     1.|
7777-999-000 | Schraube    | 006.0  **      3.00 **  :     2.|
1111-333-666 ! Zange       ! 006.0  **      9.00 **  :     3.|
9999-888-222 | Mutter      | 002.0  **      0.10 **  :     4.|
4444-555-111 ! Schlauch    ! 009.0  **     13.50 **  :     5.|
2222-444-000 | Scheibe     | 011.0  **      2.75 **  :     6.|
6666-777-222 | Nippel      | 011.0  **      4.95 **  :.    7.|
               ...                  ...
6666-111-111 ! Reifen      ! 020.0  **    468.00 **  :    23.|
**** Dateiende ****                                          |
                                                             |
                                                             |
Loading A:TURBO.MSG                                          |
Loading C:XDBASE.PAS                                         |
>_                                                           |
\____________________________________________________________/
```

Bild 5 Auflisten der dDBASE-Datensätze

5

bereich des Betriebssystems mit der Marke **KALTST** eingetragen und durch ein NUL-Byte abgeschlossen (Bild 2). Diese Festlegung lehnt sich an die Vereinbarung der Zeichenketten für Systemmitteilungen an und ermöglicht es, die Zeichenkettenlänge unmittelbar beim Transport in den Kommandopuffer zu ermitteln, um sie anschließend auf dem dafür vorgesehenen Speicherplatz abzulegen.
Als Kaltstartkommando kann jedes auf Betriebssystemebene zulässige Kommando verwendet werden. Die Autoren empfehlen die Nutzung des Kommandos SUBMIT START, das die Abarbeitung der Submitdatei START.SUB bewirkt /1/. Dadurch kann einerseits eine ganze Folge von Betriebssystemkommandos ohne Eingriff des Nutzers abgearbeitet werden, andererseits läßt sich diese Kommandofolge einfach und schnell mit Hilfe des Texteditors verändern. Fehlen auf der Diskette entweder das Programm SUBMIT.CMD oder die Kommandodatei START.SUB oder beide, so gelangt der Rechner nach einer entsprechenden Fehlermeldung in die Eingabebereitschaft des Betriebssystems. Es kann dann ohne weitere Einschränkungen gearbeitet werden.

Automatisches Einrichten der Speicherdiskette

Das Einrichten der Speicherdiskette ist an vier Voraussetzungen geknüpft:
a) Im Diskettenparameterblock (Datei DPB.LIB, s. u.) sind fünf Laufwerke definiert.
b) Das fünfte Laufwerk (logische Laufwerksnummer 4) ist als K5600.20 vereinbart.
c) Die Größe des verfügbaren Hauptspeichers beträgt mindestens 512 KByte.
d) Die Frage **DO YOU WANT A MEMORY-DISK E:...** beim Systemstart wird mit **Y** beantwortet.
Den Programmteil zum Einrichten der Speicherdiskette zeigt Bild 3. Soll darauf grundsätzlich verzichtet werden, hat man mehrere Möglichkeiten: entweder wird der gesamte Programmteil gelöscht, oder aber die Voraussetzung a) wird beim Aufbau des Diskettenparameterblockes nicht erfüllt (von den Autoren wurde nur die zweite Variante praktisch erprobt).
Soll die Speicherdiskette in jedem Fall eingerichtet werden, müssen zunächst einmal die Voraussetzungen a) bis c) erfüllt sein. Der Programmteil zur Ausgabe der Frage **DO YOU WANT A MEMORY-DISK...** und zur Auswertung der Antwort des Bedieners kann aus dem BIOS einfach herausgelöscht werden (mit * gekennzeichnete Programmzeilen im Bild 3). Etwas problematischer ist der Fall bei der zweiten Frage **INITIALIZE MEMORY-DISK...** Die Speicherdiskette muß nach dem Einschalten des Gerätes installiert werden, andererseits soll aber die Möglichkeit des Speichererhalts, auch der RAM-Diskette nach RESET /2/, erhalten bleiben. Dazu bietet sich folgende Eigenschaft des A7100 an: Nach dem Netzeinschalten oder nach RESET ohne nachfolgendem **BREAK nach TONE** erfolgt die Initialisierung des gesamten Hauptspeichers /3/, wobei das erste von der Speicherdiskette belegte Speicherwort die Information 5AA5H erhält. Beim Initialisieren der RAM-Diskette wird diese Information mit E5E5H überschrieben. Da die Wahrscheinlichkeit, bei der normalen Arbeit auf genau diesem Speicherwort die Information 5AA5H zu speichern, außerordentlich gering ist, kann man diesen Speicherinhalt als Anzeichen dafür interpretieren, daß eine Initialisierung des Hauptspeichers erfolgt ist und damit die Speicherdiskette zur weiteren Nutzung ebenfalls initialisiert werden muß. Folglich kann auch die o. g. zweite Frage mit den zur Auswertung der Antwort benötigten Programmschritten gelöscht werden (im Bild 3 mit + gekennzeichnet). Statt dessen sind die Im Bild 3 mit ! gekennzeichneten Zeilen einzufügen. Da außerdem eine Information des Nutzers wünschenswert ist, wird die Zeichenkette mit der Marke ELDSK1 (Bild 2) entsprechend geändert und auf dem Bildschirm ausgegeben (mit ? gekennzeichnete Zeilen im Bild 3).
Bei dem so geänderten Betriebssystem ist zum Erhalt des Inhaltes der Speicherdiskette nach RESET lediglich die Nutzung der Funktion **BREAK nach TONE** erforderlich /2/, /3/.

Betriebssystem mit dem Diskettenformat 780 KByte

Schon an dieser Stelle soll darauf hingewiesen werden, daß die Laderoutine des Monitorprogrammes das Diskettenformat 624 KByte erwartet /3/, so daß ein Systemstart von einer 780 K-Diskette in der vom PC 1715 bekannten Weise nach dem bisherigen Erkenntnisstand der Autoren nicht möglich ist. Die Tatsache, daß das Betriebssystem beim A 7100 vollständig im Hauptspeicher verbleibt, ermöglicht aber dennoch die problemlose Arbeit mit vom *Hausformat* abweichenden Diskettenformaten in beiden Laufwerken, solange man ohne **RESET** auskommt (was, nicht zuletzt auf Grund des relativ langwierigen Ablaufes bis zur Wiederherstellung der Arbeitsbereitschaft, angestrebt werden sollte).
Die Diskettenparameter werden im Diskettenparameterblock (Datei DPB.LIB) festgelegt. Diese Datei wird beim Assemblieren der BIOS-Quelle durch den Assembler eingefügt. Wie sie erzeugt werden kann, ist in /4/ beschrieben. Zu beachten ist lediglich, daß zur Erzeugung des Formates 624 KByte der Laufwerkstyp „SA460" anzugeben ist /2/. Soll die Speicherdiskette genutzt werden, sind die Definitionsanweisungen für 5 Laufwerke (Laufwerksnummern 0...4) anzugeben, Laufwerk 4 ist als K 5600.20 zu definieren.
Bild 4 zeigt als Beispiel den erzeugten Diskettenparameterblock für das Laufwerk B: (Laufwerksnummer 1). Gleichzeitig sind in Bild 4 die Änderungen eingetragen, die erforderlich sind, um dieses Laufwerk auf das Format 780 KByte (DS, 80 Spuren, 5 × 1024 Byte/Spur) einzustellen. Analog kann auch das Format für Laufwerk A: geändert werden. Da die Datei DPB.LIB das Assembler-Quellformat hat, können die Änderungen mit dem Texteditor vorgenommen werden.
Wurden die Parameter für beide Laufwerke geändert, ist vor der Laufwerksinitialisierung im BIOS eine Aufforderung zum Einlegen einer Diskette mit dem eingestellten Format in das Laufwerk, von dem aus das Betriebssystem geladen wurde, einzufügen; die Programmfortsetzung sollte erst nach Quittung des Bedieners erfolgen. Ein Vorschlag dazu ist in Bild 5 dargestellt.
Verbleibt eines der Laufwerke im Hausformat, kann von diesem natürlich ein völlig problemloser Systemstart in der üblichen Weise erfolgen.
Die Möglichkeit der temporären Änderung der Diskettenparameter mit Hilfe des Dienstprogrammes DISKSET bleibt in jedem Fall erhalten.

„Hartes" Bildschirmrollen

Die Betriebsmodi des Bildschirmes werden beim Systemstart durch die ESCAPE-Folge in der zweiten **DB**-Anweisung im Bild 2 eingestellt. Bild 2 zeigt bereits die geänderte Programmzeile für „hartes" Rollen.

Ablauf der Systemgenerierung

Der Ablauf der Systemgenerierung bis hin zum fertigen Betriebssystem ist in /4/ ausführlich dargestellt. Zu beachten ist lediglich, daß beim Assemblieren des BIOS grundsätzlich ein Diskettenparameterblock DPB.LIB benötigt wird. Das ist auch dann der Fall, wenn keine Änderungen der Laufwerksparameter gewünscht werden.

Hinweise

Die vorstehend gezeigten Änderungsmöglichkeiten zum Betriebssystem sind in erster Linie als Anregungen zu betrachten. Sie wurden für die Version 2.2 des SCP 1700, Generierungsdatum 20. März 1987, erprobt und damit ihre Funktionsfähigkeit – sowohl einzeln als auch im Komplex – nachgewiesen. Nachteilige Auswirkungen konnten nicht festgestellt werden, der durch das Betriebssytem belegte Hauptspeicherbereich vergrößert sich nicht. Das schließt natürlich nicht aus, daß es *elegantere* Lösungen gibt.
Die vom PC 1715 bekannte Möglichkeit, verschiedene Betriebssysteme als COM-Dateien auf der Diskette abzulegen und bei Bedarf wie normale Anwenderprogramme zu starten, ist beim A 7100 nicht ohne weiteres gegeben. Gründe dafür sind die gewählte Form der Speicherverwaltung und der Speicherschutz. Deshalb sollten angepaßte Betriebssysteme nur in dem Umfang erzeugt und genutzt werden, wie das zur Vereinfachung der Arbeit unbedingt erforderlich ist. Bei Änderungen im Betriebssystem, insbesondere der Diskettenformate, sollten in jedem Fall entsprechende Mitteilungen auf dem Bildschirm ausgegeben werden, da sonst für den Bediener sehr leicht unklare Situationen entstehen.

Literatur

/1/ Arbeitsplatzcomputer A 7100, Anleitung für den Bediener, Heft „Dienstprogramme"
/2/ Datei DOKERG.TXT auf der Systemdiskette
/3/ Arbeitsplatzcomputer A 7100, Betriebsdokumentation, Band 1 (Rechner und Geräte)
/4/ Arbeitsplatzcomputer A 7100, Anleitung für den Systemprogrammierer, Heft „Steuerprogramm SCPX"

KONTAKT

VEB IFA Getriebewerke Brandenburg, Abt. LOP, Michael Herse, Wilhelm-Bahms-Straße, Brandenburg, 1800; Tel. 573456

```
;
; TRY TO INITIATE AN ELECTRONIC DISK
;
        CMP     NUNIT,5             ;ELDISK GENERATED ?
        JL      LL8                 ;IF L NO
        MOV     CL,4                ;IS ELDISK A K5600.20 ?
        CALL    CADPB
        MOV     BX,AX
        ADD     BX,17
        MOV     AL,138
        CMP     [BX],AL
        JNZ     LL8                 ;IF NZ NO
        CMP     MEM_LENGTH,8000H    ;512 K OR MORE ?
        JB      LL8                 ;IF B NO
*       MOV     BX,OFFSET ELDSK0
*       CALL    PMSG
*       CALL    CONIN
*       AND     AL,0DFH
*       CMP     AL,59H              ;Y ?
*       JNZ     LL8                 ;IF NZ NO
*       MOV     CL,AL
*       CALL    CONOUT
        MOV     CX,SEG_LENGTH
        SUB     CX,4F80H            ;SUB 318 K
        MOV     SEG_LENGTH,CX       ;SEGLENGTH FOR USER WITHOUT DSK
        MOV     CX,MEM_LENGTH
        SUB     CX,4F80H            ;SUB 318 K
        MOV     DISK_SEG,CX         ;STARTING ADDRESS OF DISK
+       MOV     BX,OFFSET ELDSK1
+       PUSH    CX
+       CALL    PMSG
+       CALL    CONIN
+       POP     CX
+       AND     AL,0DFH
+       CMP     AL,59H              ;Y ?
+       JNZ     LL8                 ;IF NZ NO
+       PUSH    CX
+       MOV     CL,AL
+       CALL    CONOUT
+       POP     CX
        PUSH    DS
        MOV     DS,CX
        MOV     CX,2000H
        XOR     BX,BX
!       CMP     [BX],5AA5H      ;Hauptspeicher initialisiert?
!       JNE     LL9             ;nein
        MOV     AX,0E5E5H
LL4:    MOV     [BX],AX
        ADD     BX,2
        LOOP    LL4
LL9:    POP     DS              ;Marke LL9 einfuegen!
?       MOV     BX,OFFSET ELDSK1 ;geaenderte Information
?       CALL    PMSG            ;Ausgabe einer Mitteilung an den
                                ;Bediener
```

Bild 3 *Programmteil zum Einrichten der Speicherdiskette (Auszug aus der Datei BIOS.A86 mit Änderungen)*

```
                Parameter fuer Diskettenformat

                640 K        780 K
                (erzeugt)    (zu aendern)
;
DPB1    EQU     OFFSET $                  ;Disk Parameter Block
        DW      64           80           ;Sectors per Track
        DB      4                         ;Block Shift
        DB      15                        ;Block Mask
        DB      0                         ;Extent Mask
        DW      311          389          ;Disk Size - 1
        DW      127                       ;Directory Max
        DB      192                       ;Alloc0
        DB      0                         ;Alloc1
        DW      32                        ;Check Size
        DW      2                         ;Offset
        DB      1            3            ;Physical Sec Shift
        DB      1            7            ;Physical Sec Mask
        DB      139                       ;Drive Descriptor
        DW      80                        ;Cylinders
        DB      0                         ;Fixed Heads
        DB      2                         ;Movable Heads
        DB      16           5            ;Phys Sectors per Track
        DW      256          1024         ;Bytes per Sector
        DB      1                         ;Reserved Cylinders
XLT1    EQU     0                         ;No Translation Table
ALS1    EQU     50                        ;Allocation Vector Size
CSS1    EQU     32                        ;Check Vector Size
```

Bild 4 *Diskettenparameterblock für Laufwerk B: (Auszug aus Datei DPB.LIB mit Änderungen für Format 780 KByte)*

Lichtwellenleiter kontra CSMA/CD?

Andreas Barsch
Humboldt-Universität zu Berlin, Sektion Elektronik

Andreas Barsch (30) studierte von 1979 bis 1984 an der Sektion Elektronik der Humboldt-Universität zu Berlin. Er diplomierte dort mit einer Arbeit auf dem Gebiet der LAN-Controller (LANCELOT 1). Gegenwärtig befaßt er sich als wissenschaftlicher Assistent mit Hardwareaufgaben für schnelle lokale Netze mit optoelektronischer Informationsübertragung (LANCELOT 2).

Der Artikel befaßt sich mit der Problematik der Intergration des modernen und vorteilhaft einsetzbaren Übertragungsmediums Lichtwellenleiter in CSMA/CD-LANs. Ausgehend von einer kurzen Analyse des Zugriffsverfahrens CSMA/CD, werden durch beispielbezogene topologische Betrachungen vor allem Möglichkeiten der Kollisionserkennung untersucht. Der Aspekt einer völligen Mediumunabhängigkeit, bezogen auf das Zugriffsverfahren, spielt eine besondere Rolle.

Vorbemerkungen

Lokale Netzwerke, die in der Verbindungsschicht ein CSMA/CD-Zugriffsverfahren aufweisen, haben trotz alternativer Methoden im Bereich der Bürokommunikation kaum an bedeutung verloren. Ursachen dafür sind insbesondere der vergleichbar geringe Steuerungsaufwand bei der Implementation und die Toleranz gegenüber Stationsausfällen.
Auf der Suche nach modernen Übertragungsmedien für lokale Netze befindet sich der Lichtwellenleiter gegenwärtig in favorisierter Position. Seine große Übertragungsbandbreite, die absolute Immunität gegenüber elektromagnetischen Einflüssen, aber vor allem die Möglichkeit der galvanischen Entkopplung der Kommunikationspartner, die Kupfereinsparung, die Gewichtsreduzierung und nicht zuletzt die schlechte Abhörbarkeit des Lichtwellenleiters sind entscheidende Vorteile gegenüber anderen Übertragungsmedien. Trotz der gegenwärtig noch höheren Anschlußkosten für den Lichtwellenleiter ergibt sich aufgrund der genannten Vorteile die Möglichkeit und Notwendigkeit der Integration optischer Komponenten in CSMA/CD-Netze. Dabei sei es bei der folgenden Betrachtung der technischen Hintergründe zunächst ohne Bedeutung, ob der Aufbau eines homogenen optischen CSMA/CD-LANs oder die Substitution einzelner Komponenten durch optische Elemente angestrebt wird. Wesentlich ist vielmehr die Tatsache, eine völlige Mediumunabhängigkeit, bezogen auf das Buszugriffsverfahren, und damit die Einhaltung aller Implementationsbedingungen eines betrachteten Netzwerkes zu erreichen. Dazu ist die Frage zu klären:

Wie funktioniert CSMA/CD?

Der Name des Verfahrens CSMA/CD (**C**arrier **S**ense **M**ultiple **A**ccess with **C**ollision **D**etection) weist zunächst darauf hin, daß in Abhängigkeit vom Belegungszustand der Übertragungskanal von allen angeschlossenen Netzrechnern (NR) gleichberechtigt benutzt werden kann. Aufgrund dieses einzigen Benutzungskriteriums kann es durch die Signallaufzeiten auf dem Kanal zu unerwünschten Mehrfachbenutzungen kommen, wenn mehrere NR gleichzeitig einen anhängigen Sendewunsch realisieren möchten. Das führt in jedem Fall zu einer unkorrekten Übertragung der gesendeten Daten. Um eine derartige Situation möglichst schnell erkennen und auflösen zu können, wird die Funktion „Collision Detection" (Kollisionserkennung) genutzt. Hiermit ist es möglich, die Sendung definiert abzubrechen und zu einem späteren Zeitpunkt zu wiederholen. Diese globale Beschreibung trifft das Wesen aller CSMA/CD-Verfahren. Einzelheiten sind von den konkreten Implementationsbedingungen abhängig. Gegenwärtig existiert eine ganze Reihe von CSMA/CD-Netzwerken mit unterschiedlichen Eigenschaften (Tafel 1) /1/.
Grundanliegen eines jeden lokalen Netzes ist es, eine möglichst sichere Datenübertragung schon im Bereich der unteren Schichten zu erreichen. Das impliziert bei CSMA/CD-Netzen das eindeutige Erkennen von Kollisionssituationen durch alle sendenden NR und das Verwerfen von Blockfragmenten (Blocklänge kleiner als minimale Blocklänge) durch alle empfangenden NR. Insbesondere die zweite Tatsache zwingt zur strikten Beach-

Tafel 1 ***Überblick über CSMA/CD-LANs***

LAN	*Firma*	*DR*	*Topologie*	*Medium*
ETHERNET	DEC, XEROX, INTEL	10	Linienbus	Koax
EtherLink	3Com	10	Linienbus	Koax
Net/One	Ungermann/Bass	10	Linienbus	Koax
PC Network	IBM	2	Linienbus	BB
G-Net	Gateway-Comm.	1,4	Linienbus	Koax
PC net	Orchid	1	Linienbus	Koax
Omninet	Corvus	1	Linienbus	TP
StarLan	AT&T	1	Linienbus	TP
ROLANET1	Robotron	0,5	Linienbus	Koax
PC Cluster	IBM	0,375	Linienbus	Koax

DR = Datenrate in MBit/s
TP = twisted pair
BB = Breitbandübertragung

Bild 1 Verzögerungszeiten im Übertragungskanal

tung des Grundzusammenhanges von maximaler Systemausdehnung (topologischer Faktor), der damit verbundenen maximalen Signallaufzeit auf dem Übertragungskanal und der Datenübertragungsgeschwindigkeit (übertragungstechnischer Faktor).

NR1 und NR2 seien die am weitesten entfernten Stationen in einem lokalen Netz (Bild 1). Dann kommen die von NR1 ausgesendeten Daten um die Zeit $t_{D1,2}$ verzögert bei NR2 an, wobei $T_{D1,2}$ alle im Signalfluß auftretenden Verzögerungen umfaßt (auch Transceiver, Repeater ...). Während dieser Zeit stellt NR2 trotz Belegung durch NR1 einen freien Übertragungskanal fest und kann diesen selbst mit zu sendenden Daten belegen. Die so möglicherweise entstandene Kollision wird durch NR2 relativ schnell detektiert. NR1 hingegen kann diesen Zustand erst nach $T_{D2,1}$, der Verzögerungszeit der rücklaufenden Signale, feststellen, um nach einer kurzen Wartezeit t_{DC} (zum Forcieren des Kollisionszustandes) die Sendung abzubrechen. Für maximal die Zeit

$$t_{TX} = T_{D1,2} + T_{D2,1} + t_{DC} \quad (1)$$

ist der Übertragungskanal also belegt. Ein empfangsbereiter NR sammelt die Daten folglich während dieser Zeit und interpretiert sie als Block. Bei zufälliger Übereinstimmung der letzten empfangenen Bits mit dem im Empfänger berechneten CRC wird aus den kollidierten Daten ein richtig empfangener Block. Um diese Möglichkeit auszuschließen, prüft ein empfangender NR zuvor, ob die Anzahl der gesammelten Bits größer ist als die, die während der Zeit t_{TX} maximal empfangen werden kann, d. h. ob der Block länger ist als die minimale Blocklänge (MBL). In der MBL kommt der bereits erwähnte Zusammenhang von topologischem Faktor und übertragungstechnischem Faktor zum Ausdruck:

$$MBL\,[Byte] > (DR\,[Mbit/s] * t_{TX}\,[\mu s])/8 \quad (2)$$

Auf diese Weise wird es möglich, durch Kollision entstandene und richtig übertragene Blöcke sicher zu unterscheiden und gegebenenfalls von der Weiterverarbeitung auszuschließen. Die MBL hat aber zugleich einen nicht zu unterschätzenden Einfluß auf die Gesamteffizienz des Netzes. Jede Sendung belegt mindestens für die Zeit t_{TX} den gemeinsamen Übertragungskanal, unabhängig davon, wieviele Bytes wirklich zu senden waren. Es gilt also, einen geeigneten Kompromiß zu finden.

Internationale Standardisierungsbemühungen für ein einheitliches CSMA/CD-LAN fanden u. a. in /2/ ihren Niederschlag. Die hier und in /3/ festgelegten Implementationsbedingungen lösen den genannten Kompromiß. Für ein koaxialkabelgebundenes LAN mit einer Ausdehnung von maximal 2500 Metern gelten folgende zeitliche Bedingungen:

$t_{PD} = 44{,}99\,\mu s = t_{D1,2} + t_{D2,1}$ (max. Umlaufverzögerung)
$t_{JAM} = 3{,}2 \text{ bis } 4{,}8\,\mu s = t_{DC}$ (JAM-Time)
$t_{SLOT} = 51{,}2\,\mu s > t_{PD} + t_{JAM}$ (SLOT-TIME).

Daraus ergibt sich bei der festgelegten Datenrate von DR = 10 Mbit/s eine minimale Blocklänge von

$$MBL = (t_{SLOT} * DR) / 8 = 64 \text{ Byte} \quad (3)$$

Im Gegensatz dazu darf der größte Block maximal 1518 Bytes enthalten.

Die SLOT-Time bestimmt nicht nur die maximale Ausdehnung des Netzes und die minimale Blocklänge, sondern wird auch als Quantifizierungsfaktor für den sogenannten Retransmissions-Algorithmus verwendet, der in Abhängigkeit von der Sendeversuchszahl den Zeitpunkt für eine erneute Sendung nach einer Kollision bestimmt. Ziel dabei ist es, eine kollidierte Sendung möglichst schnell zu wiederholen, aber auch eine erneute Kollision möglichst sicher zu vermeiden. Der Abstand der neu berechneten Zeitpunkte in den konkurrierenden NR sollte demzufolge mindestens t_{PD} betragen. In der Praxis wird die SLOT-Time t_{SLOT} benutzt, deren wesentlichster Bestandteil ja die Umlaufverzögerungszeit t_{PD} ist.

Der in /2/ festgelegte Algorithmus verwendet eine Zufallszahl R zur Berechnung von t_{RETR}, die im Intervall $0 <= R < 2^K$ liegt. Die Variable k ist abhängig von der Anzahl der Sendewiederholungen n, wobei nach der fünfzehnten erfolglosen Wiederholung der Algorithmus abgebrochen wird. Da für das betrachtete LAN maximal 1024 Stationen zugelassen sind, ist die Verteilung von mehr als 1024 Retransmissions-Zeitpunkten nicht sinnvoll. Aus diesem Grund ist k auf maximal 10 begrenzt. Für k gilt also:

$$k = \text{Min}\,(n, 10) \quad (4).$$

Die Zeit t_{RETR} ergibt sich letzlich zu:

$$t_{RETR} = R * t_{SLOT} \quad (5).$$

Das sichere Vermeiden einer erneuten Kollision kann insbesondere aufgrund des zufälligen Wertes R in allen NR nicht erreicht werden.

Kollisionserkennung – optimale Topologie

Aus dem bisher Dargelegten ist erkennbar, daß sich im wesentlichen zwei technische Probleme beim Übergang zu optischen Übertragungskanälen unter dem Aspekt einer völligen Mediumunabhängigkeit in einem CSMA/CD-LAN ergeben. Erstens muß die für das betrachtete LAN spezifizierte maximale Umlaufverzögerungszeit t_{PD} dahingehend eingehalten werden, daß spätestens nach dieser Zeit jedem sendenden NR eine aufgetretene Kollision mitgeteilt wird. Das ist in gewissen Grenzen relativ einfach möglich, da die Signallaufzeiten auf dem Lichtwellenleiter etwa denen auf dem Koaxialkabel entsprechen, eine Abhängigkeit also nur von topologischen Faktoren (Struktur, Ausdehnung) besteht. Wesentlich komplexer präsentiert sich hingegen ein zweites Problem, die dezentrale Kollisionserkennung durch die NR in optischen LANs selbst. Jede Kollision weist in Abhängigkeit von der spezifischen Topologie und dem verwendeten Übertragungsmedium des betrachteten LANs ganz typische Eigenschaften, die sogenannten Basismerkmale auf. So kann man in elektrischen LANs mit Linienbus im wesentlichen drei Basismerkmale bei Kollisionen feststellen:

1. Die gesendeten Daten stimmen nicht mit den vom Bus empfangenen überein, da eine Überlagerung von elektrischen Impulsen stattgefunden hat.
2. Die Bildungsvorschrift des verwendeten Leitungskodes (meist Manchester-Code) wird durch eine Überlagerung verletzt.
3. Schwebungseffekte können mittels Pegelbewertung erkannt werden.

Durch eine entsprechende hardwaretechnische Umsetzung in jedem NR, insbesondere unter Verwendung der Punkte 1 und 3, ist eine einfache dezentrale Kollisionserkennung möglich.

Betrachtet man jetzt einige prinzipielle topologische Möglichkeiten optischer LANs vor allem unter dem Aspekt der Kollisionserkennung, aber auch unter dem Aspekt der technischen Realisierbarkeit, so ergibt sich ein sehr differenziertes Bild.

Der **optische Linienbus** (Bild 2) als naheliegendste Struktur und in Anlehnung an das elektrische Vorbild hat gegenwärtig keine praktische Bedeutung. Ursache dafür sind die mit der passiven Ankopplung verbundenen hohen Dämpfungsverluste, so daß mit den verfügbaren Mitteln (T- und Y-Koppler) keine größeren Netzwerke ohne Zwischenverstärkung realisierbar sind. Die Möglichkei-

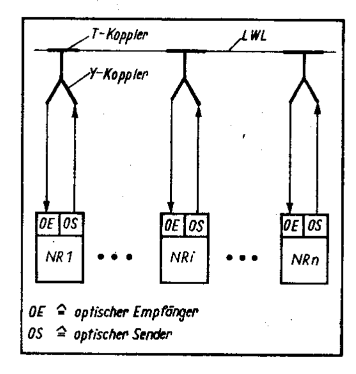


Bild 2 Linienbus mit passiver Ankopplung

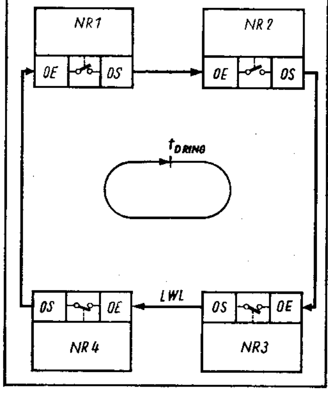


Bild 3 Ringbus mit aktiver Ankopplung

ten bei der Kollisionserkennung gleichen etwa denen beim elektrischen Vorbild. Durch die gleichzeitige Mehrfachbenutzung des Übetragungskanals kommt es zur Überlagerung von Lichtimpulsen. Die resultierenden Basismerkmale ergeben sich wie folgt:

1. Die gesendeten Daten stimmen nicht mit den vom Bus empfangenen Daten überein.

2. Die Bildungsvorschrift des verwendeten Leitungskodes wird verletzt.

3. Schwebungseffekte können durch Pegelbewertung erkannt werden (Empfänger mit Analogausgang notwendig).

Die praktische Nutzbarkeit der genannten drei Basismerkmale ist stark von den dynamischen Verhältnissen im Gesamtsystem abhängig (insbesondere Merkmal 3). Prinzipiell ist eine sichere dezentrale Kollisionserkennung durch Basismerkmalsanalyse möglich.

Der **optische Ringbus** mit aktiver Ankopplung (Bild 3) ist dadurch gekennzeichnet, daß ein sendender NR den Ring mittels einer entsprechenden Schaltvorrichtung (Repeater) auftrennt und ihn erst beim Empfang von Fremddaten während einer eigenen Sendepause wieder schließt. Im Gegensatz zum Linienbus wird hier der Lichtwellenleiter unidirektional genutzt. Im kollisionsfreien Fall gelangen die Daten vom sendenden NR (z. B. NR1) über den Ring zum Ziel-NR (z. B. NR4) und liegen nach der Ringumlaufzeit t_{DRING} unverändert wieder beim sendenden NR an. Die Realisierung eines weiteren Sendewunsches (z. B. NR2 an NR3) zum gleichen Zeitpunkt führt dazu, daß beide sendenden NR (NR1 und NR2) den Ring auftrennen und damit ein Weiterleiten von ankommenden Daten verhindern. NR3 empfängt zwar die Nachrichten von NR2, die von NR1 erreichen aber nicht NR4. Es ist zu erkennen, daß eine Kollision im Sinne einer Überlagerung von Daten unter den gegebenen Bedingungen nicht auftritt. Als einziges Basismerkmal läßt sich daher konstatieren, daß die gesendeten Daten nicht immer den Ziel-NR, aber nie den Ursprungs-NR erreichen. Mittels eines Koinzidienz-Sensors im sendenden NR ist ein Vergleich von gesendeten und empfangenen Daten zur Detektion eines Kollisionszustandes möglich, wobei die implementationsabhängige Zeitdifferenz t_{DRING} zwischen den beiden Datenströmen zu erheblichen technischen Schwierigkeiten führt. Die Einhaltung der maximalen Signalumlaufzeit t_{PD} erscheint bei Anschluß einer größeren Anzahl von NR ebenfalls problematisch, da die meisten optischen Empfänger mit Filtern ausgerüstet sind, die eine Signalweiterleitung erst nach dem Erkennen von einigen gültigen Datenbits ermöglichen. Die Summe der Signallaufzeiten überschreitet dadurch schnell das vorgegebene Limit. Auch aufgrund weiterer Probleme (z. B. Bypass-Verhalten der Repeater) erscheint eine aktive optische Ringstruktur unter Benutzung eines CSMA/CD-Zugriffsverfahrens ohne praktische Bedeutung.

Der **optische Sternbus** mit einem zuführenden und einem wegführenden Lichtwellenleiter als Verbindungsglied zwischen Sternkoppler und NR (Bild 4) sei als weitere topologische Möglichkeit betrachtet. In Abhängigkeit davon, ob im Sternkoppler elektrisch-optische bzw. optisch-elektrische Wandler eingesetzt sind oder nicht, unterscheidet man aktive und passive optische Sternkoppler. Wesentlichster Unterschied zu allen bereits betrachteten topologischen Strukturen ist

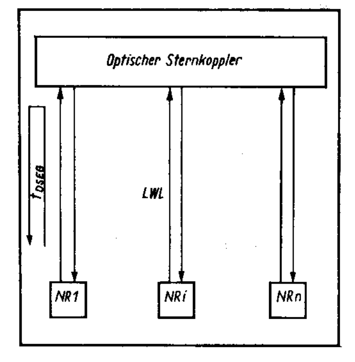


Bild 4 Sternbus

das Zusammentreffen aller Datenströme in einem zentralen Punkt, dem Sternkoppler. Damit eröffnen sich für das Gesamtproblem der Kollisionserkennung völlig neue Möglichkeiten – die *Transformation von Basismerkmalen* in dezentral günstiger auswertbare Zweitmerkmale (ZMM). Verzichtet man auf die Transformation im Sternkoppler, können folgende Basismerkmale zur dezentralen Auswertung in den NR genutzt werden:

1. Die empfangenen Daten stimmen aufgrund der Überlagerung im Sternkoppler nicht mit den gesendeten Daten überein.

2. Die Bildungsvorschrift des Leitungskodes wird verletzt.

Da die Kontrolle der Kodierungsvorschrift (z. B. nach /4/) in Abhängigkeit von der Anzahl der NR sehr aufwendig ist und vor dem Vergleich von gesendeten und empfangenen Daten das technische Problem des Ausgleichs der längenabhängigen Segmentumlaufzeit t_{DSEG} steht, stellt die Transformation von Basismerkmalen eine echte Alternative dar. So können in einem aktiven Sternkoppler die existierenden Basismerkmale wie

1. Verletzung der Bildungsvorschrift des Leitungskodes

2. gleichzeitige Belegung mehrerer Kopplerkanäle

detektiert und in Zweitmerkmale transformiert werden. Das Senden eines Signals mit einer anderen Frequenz /5/ oder einer anderen optischen Wellenlänge /6/ auf der Empfangsleitung des NR oder das Rückkoppeln eines Signals auf der bidirektionalen Sendeleitung des NR /6/ sind derartige Zweitmerkmale. Weiterhin kann durch einen Sternkoppler, der die ankommenden Signale auf alle wegführenden Lichtwellenleiter verteilt, eine Kollision dadurch gekennzeichnet werden, daß für die sendenden NR die Reflexion abgeschaltet wird. Ein sendender NR empfängt also für die Dauer der Kollision keine Signale (ZMM = Echoverlust während der Sendung). Die entgegengesetzte Möglichkeit bietet sich durch einen Sternkoppler, der im Grundzustand ein ankommendes Signal auf alle wegführenden Lichtwellenleiter verteilt, nicht aber auf den zum signalzuführenden Lichtwellenleiter korrespondierenden. Hier wird im Kollisionsfall für die sendenden NR die Reflexion eingeschaltet (ZMM = Echo während der Sendung). Es ist zu erkennen, daß sich mittels eines aktiven optischen Sternkopplers eine Vielzahl von Transformationsmöglichkeiten ergeben, die zu einer sicheren dezentralen Kollisionserkennung genutzt werden können. Aus diesem Grund ist die Sternstruktur als favorisierte optische Topologie für den Einsatz von CSMA/CD-Buszugriffsverfahren zu betrachten. Die Auswahl eines geeigneten Zweitmerkmals als Transformationsziel ist von vielen technischen und ökonomischen Randparametern abhängig, u. a. von der Anzahl der benötigten Kanäle und davon, ob ein homogenes optisches LAN zu konzipieren ist und ob Kaskadierungsmöglichkeiten vorzusehen sind. Auf jeden Fall müssen die erhöhten Zuverlässigkeitsanforderungen durch die zentrale Position des Sternkopplers berücksichtigt werden.

Schlußbemerkung

Zusammenfassend kann eingeschätzt werden, daß unter Beachtung der beschriebenen Zusammenhänge die Vorzüge des optischen Übertragungsmediums durchaus auch in CSMA/CD-Netzen entfaltet werden können. Homogene optische LANs auf der Basis aktiver Sternstrukturen bilden eine Form der Nutzung. Eine nicht zu unterschätzende Praxisrelevanz kommt als zweite Nutzungsform der Substitution von elektrischen Komponenten zu. Ziel dabei ist es, die Einsatzbreite von existierenden kabelgebundenen CSMA/CD-LANs mit optimalen Mitteln zu erweitern und neue Einsatzfälle zu erschließen. Unter Beachtung der neuen Randbedingungen ist die Nutzung der beschriebenen Wirkmechanismen auch für diesen Fall möglich.

Literatur

/1/ LAN Evolution Report 1986
1986 by NOVELL, Inc., Orem, Utah

/2/ ECMA 80, 81, 82
2. Ausgabe 1984

/3/ The Ethernet, A Local Area Network, Specification, Version 1.0, 1980

/4/ Barsch, A. u. a.: Schaltungsanordnung zur Dekodierung und Fehlererkennung optisch übertragener Binärsignale, DD-WP 214979, 1983

/5/ Schmidt, R. V. u. a.: Fibernet II: Fiber Optic Ethernet IEEE Selected Areas in Communications Nov. (1983) Vol. SAC-1, No. 5

/6/ Witte, H. H.: Optisches Sternnetz für stochastische Zugriffsverfahren, DE-OS 3435827, 28. 9. 84

✉ KONTAKT ☏

Sektion Elektronik, Humboldt-Universität zu Berlin, Bereich Technische Informatik, Invalidenstr. 110, Berlin, 1040; Tel. 2 80 35 78

TERMINE

5. Computerfachtagung

WER? Bezirksvorstand Frankfurt (O.) der KDT

WANN? 26. und 27. 10. 1988

WO? Frankfurt (Oder)

WAS?

- KC 85/2 ... 4 und Peripherie
- Einchipmikrorechner
- FORTH
- neue Betriebssysteme
- Ausstellung „Computer der volkseigenen Industrie, Eigenbaucomputer und Softwareentwicklung"

WIE? Meldungen für die Ausstellung bis 30. 8. 1988 an Dr. Scheuschner, Ingenieurbetrieb Mikroelektronik Frankfurt (O.), Tel.: 36 92 42. Nähere Informationen sind beim BV Frankfurt (O.) der KDT, Ebertusstr. 2, Tel.: 36 93 60, erhältlich.

Georgi

PC-1715-Funktionstastenbelegung durch Anwenderprogramme

Bernd Matzke, Delitzsch

Der PC 1715 besitzt Funktionstasten, deren Inhalt durch den Anwender festgelegt werden kann. Normalerweise kann deren Initialisierung nur bei der Installierung des Betriebssystems erfolgen. Mitunter ist es jedoch wünschenswert, die Funktionstasten mit in bestimmte Anwenderprogramme einzubeziehen oder ihnen häufig gebrauchte Schlüsselwörter zuzuweisen. Im folgenden wird der Aufbau der Code-Tabelle für die Funktionstasten beschrieben und ein PASCAL-Programm vorgestellt, mit dessen Hilfe die Initialisierung der Funktionstasten erfolgen kann. Das Programm wurde unter dem Betriebssystem SCPX-1715 Vers. 0.4 und Vers. 0.5 erprobt.

Aufbau der Code-Tabelle

Bei einem im Hauptspeicher residenten, also aktiven, Betriebssystem beginnt die Code-Tabelle ab Adresse E0AAH. Sie ist 64 Byte lang. Die Informationen für jede Tastenbelegung sind in Form von einzelnen Blöcken aneinandergereiht. Dabei besitzt ein einzelner Block folgenden Aufbau:

- 1. Byte : Code der Taste
- 2. Byte : Länge des zuzuweisenden Textes
- ab 3. Byte : Text (in ASCII-Codierung).

Im Text können alle ASCII-Zeichen enthalten sein. Schließt man z. B. einen Text mit dem Code für „ET“ ab (0DH), so wird bei Betätigen der entsprechenden Taste der Text und Return ausgegeben, das entsprechende Kommando also sofort ausgeführt.

Beschreibung des Programms

Das Programm zur Initialsierung der Funktionstasten (FUTAST) wurde in TURBO-PASCAL erstellt. Es liest die gewünschte Tastenbelegung aus einer Textdatei (TASTCODE.TXT) aus und weist den Inhalt der einzelnen Zeilen den Funktionstasten in der Reihenfolge F1 bis F14 zu. Das Unterstreichungszeichen ("_") wird dabei in den Code für <ET> gewandelt.

Sind in der Datei weniger als 14 Zeilen enthalten, werden die entsprechenden Funktionstasten nicht belegt. Ebenso wird einer Taste kein Text zugewiesen, wenn die entsprechende Zeile leer ist, sie also nur die Codes für Wagenrücklauf und Zeilenvorschub enthält. Überzählige Zeilen werden übersprungen.

Die Datei TASTCODE.TXT kann von beliebigen Anwenderprogrammen erzeugt werden. Beim Erstellen mit dem Textprozessor ist sie als Programmdatei zu erzeugen (N-Option).

Als praktisches Beispiel soll die Initialisierung der Funktionstasten in einem REDABAS-Programm dienen. Durch geringfügige Änderung des PASCAL-Programmes kann der Name der Textdatei vom Nutzer eingegeben werden. Damit ist es möglich, unter mehreren vorbereiteten Textsätzen für die Funktionstasten auszuwählen.

```
program futast;
type
  code=array[1..14] of byte;
const
  ftastcode:code=($D1,$D2,$D3,$D4,$CF,$A0,$A1,$A2,$A3,$83,$C1,$C0,$C2,$CD);
  anfadr=$E0AA;
var
  filevar:text;
  line:string[64];
  adresse,i,satznr:integer;
  tastnr:byte;
begin
  adresse:=anfadr;
  repeat
    mem[adresse]:=$FF;                { Loeschen des alten Tabelleninhaltes }
    adresse:=adresse+1;
  until adresse=anfadr+64;
  adresse:=anfadr;
  satznr:=0;
  assign(filevar,'tastcode.txt');
  reset(filevar);                   { Eroeffnen des Files mit Tasten-Belegung }
  while not eof(filevar) do
  begin
    readln(filevar,line);
    satznr:=satznr+1;
    if (length(line)>0) and (satznr<15) then
    begin
      if adresse+length(line)<anfadr+64 then
      begin
        tastnr:=ftastcode[satznr];
        mem[adresse]:=tastnr;                    { Eintragen des Tasten-Codes }
        mem[adresse+1]:=length(line);                 { Eintragen Textlaenge }
        for i:=1 to length(line) do
        begin
          mem[adresse+1+i]:=ord(line[i]);                   { Eintragen Text }
          if line[i]='_' then
            mem[adresse+1+i]:=$0D;                          { Code fuer <ET> }
        end;
      end
      else
     begin
       writeln('Code-Tabelle voll');
       writeln( 'Tasten bis Nr. ',satznr-1,' belegt');
       while not eof(filevar) do
        readln(filevar,line);{Ueberspringen aller ueberzaehligen Saetze}
     end;
     adresse:=adresse+length(line)+2;
   end;
  end;
end.
```

```
* Programm Tastdemo
*
* Dieses Programm schreibt die eingegebenen Tastenbelegungen
* in die REDABAS-Datei FUTASTEN.
* Diese Datei besteht aus 14 Datensaetzen. Jeder Datensatz enthält
* ein Feld "Tastcode" (Typ C) mit einer beliebigen Laenge.
* Die Datei FUTASTEN ist vor dem Aufruf dieses Programmes selbst
* zu erzeugen !
* Anschliessend wird eine Datei "TASTCODE.TXT" im
* Standarddatenformat erzeugt.
* In ihr ist in jeder Zeile eine Tastenbelegung enthalten.
* Nach der Abarbeitung des Programms FUTAST erfolgt eine Rückkehr
* zu REDABAS.
*
set talk off
use FUTASTEN
go top
do while .not.eof
  erase
  § 2,2 say #
  § 2,7 say '.Taste' get Tastcode
  read
  clear get
  skip
enddo
copy to TASTCODE field Tastcode sdf delimited with,
quit to 'FUTAST','REDABAS'
```

TERMINE

Jahrestagung der WGMA „Automatisierungstechnik“
WER? Wissenschaftlich-Technische Gesellschaft für Meß- und Automatisierungstechnik im Präsidium der KDT
WANN? 15.–16. 12. 1988
WO? Magdeburg
WIE? Schriftliche Teilnahmewünsche an: Präsidium der KDT, WGMA, PSF 1315, Berlin, 1086

Müller



Bericht

Fachtagung „Bildanimation mit Computern“

Am 27./28. Oktober 1987 fand die 1. Fachtagung *Bildanimation mit Computern* an der Technischen Universität „Otto von Guericke“ Magdeburg statt. Veranstalter waren die Kammer der Technik, die Sektion Informatik der TU Magdeburg, das Institut für Film, Bild und Ton (ifbt) und die Nationale Vereinigung für wissenschaftlichen Film und Fernsehen (NWF) der DDR.

16 Fachvorträge stützten sich weitgehend auf Live-Demonstrationen von mit Kleinrechnern erzeugten bewegten Bildschirmbildern und informierten anschaulich über den erreichten Stand und die bearbeiteten Anwendungsfelder. Die Bildanimation mit Hilfe grafiktüchtiger Kleincomputer (hauptsächlich auf KC 85) bildete den Schwerpunkt der Fachtagung.

Zwei Übersichtsvorträge beschäftigten sich mit den derzeitigen Hauptanwendungsgebieten, dem internationalen Hard- und Softwarestand und gaben einen Überblick über die Verfahren zur Erzeugung bewegter Bilder. Dabei wies *Prof. Lorenz* (TU Magdeburg) auf die engen Beziehungen zwischen Bildanimation und Simulation hin. Bewegte Bilder sind laufend mit Bewegungsinformationen zu versorgen, Simulationsmodelle erzeugen chronologisch Zustandsinformationen und Resultatdaten, die durch Prozeßablaufgrafiken verdichtet und anschaulich darzustellen sind. Die Arbeiten an der TU Magdeburg konzentrieren sich auf die Bildanimation zur Präsentation von Simulationsmodellen und -resultaten. In einem zweiten Übersichtsvortrag über „Computervision“ erläuterte *Huth* (Zeitschrift Bild und Ton) vier Umgangsformen der Computer mit Bildern, nämlich die Bildgewinnung, die Bildverarbeitung, die Bildsimulation und die Computergrafik.

Die Animationstechnik wird an vielen Hochschuleinrichtungen bereits systematisch zur visuellen Unterstützung der Ausbildung genutzt, wie etwa in Vorlesungen und Praktika der Elektrotechnik (*Prof. Albrecht*, Universität Rostock), in Verbindung mit statistischen Prozeßanalysen (*Dr. Dreyer*, TU Dresden), zur Darstellung komplizierter räumlicher Strukturen von Atomclustern, Gitterausschnitten und Atomorbitalen aus verschiedenen Blickwinkeln (*Dr. Kadura*, Universität Jena) oder zur Demonstration von Zufallsprozessen (*Prof. Lorenz*, TU Magdeburg).

Die Animationstechnik eignet sich sowohl zur Präsentation von Simulationsresultaten, wie z. B. bei der Modellierung von Stoffflußprozessen in CAD-Entwurfssystemen für die Betriebsplanung (*Dr. Jetschny*, TU Dresden), als auch für die Information und Werbung auf Messen und Ausstellungen, wie die Präsentationsgrafik für das Kombinat ASUG zeigte (*Haas*, Zentrales Projektierungsbüro der Textilindustrie Leipzig). Ein rechnergestützter dialogorientierter Touristeninformator für die Hauptstadt Berlin nutzt ebenfalls grafische Bildelemente (*Flemming*, HfV Dresden).

Einen weiteren Schwerpunkt der Fachvorträge bildeten Softwaretools für die Animationstechnik, und zwar zur

– Beschreibung und Darstellung dreidimensionaler Objekte im Dialog (EXSY3D – KC 85/3) und Erzeugung von Bildsequenzen durch Koordinatenstransformation (*Prof. Werler*, TU Magdeburg),
– Konstruktion benutzereigener Sonderzeichen (*Görgens*, TU Magdeburg), wobei aus Folgen solcher punktweise verschobenen Sonderzeichen quasi-kontinuierliche Bewegungsabläufe erzeugt werden (*Behlau*, TU Magdeburg),
– Basissoftwareentwicklung für die Bildanimation auf SKR (*Nowak*, TU Magdeburg),
– Konstruktion, Speicherung und Manipulation von Grafiken zur Erzeugung von (Tapeten-)Mustern (*Dr. Kolbe*, Hochschule für industrielle Formgestaltung, Halle).

Vergleiche der Abarbeitungsgeschwindigkeit von Simulationsmodellen, die in BASIC, PASCAL oder FORTH geschrieben wurden, zeigten an Hand der parallelen Prozeßdarstellung und Resultatausgabe durch bewegte Bilder, daß mit PASCAL und besonders mit FORTH schnelle nebenläufige Prozesse realisiert werden können (*Buchholz/ Nitzsche/ Schmidt*, TU Magdeburg).

Im Ergebnis der Tagung kann festgestellt werden, daß

– mit Hilfe der grafiktüchtigen Kleincomputer durchaus anspruchsvolle wissenschaftliche und technische Animationsprobleme gelöst werden können
– ein beachtlicher Stand bei den konkreten Anwendungslösungen für Ausbildungszwecke, für die Präsentation von Simulationsmodellen und -resultaten und für Information und Werbezwecke zu verzeichnen ist, jedoch
– die Softwareunterstützung für den KC 85 zum dialoggestützten Bildaufbau, zur Bildverwaltung und zur Bewegungsbeschreibung und -ausführung für mehrere unabhängige Bildelemente noch nicht ausreichend entwickelt ist.

Dr. Dietrich Ziems

Productronica '87

Vom 9. bis 13. November 1987 fand in München zum 7. Mal die Productronica statt. Mit etwa 1600 Ausstellern auf 105 000 m^2 Ausstellungsfläche ist sie neben der Semicon/Zürich, der Pronic/Paris und der Elektromas/ Moskau die bedeutendste Fachmesse für die Fertigung in der Elektronik.

Die Aussteller aus 27 Staaten, darunter 21 europäische und 6 des RGW und die führenden Elektronikländer USA und Japan, zeigten mit ihren Exponaten den erreichten internationalen Stand

– bei technologischen Spezialausrüstungen (TSA) für die Herstellung von aktiven und passiven Bauelementen mit dem Schwerpunkt bei hochintegrierten (VLSI-) und SMD-Bauelementen
– bei der Kontroll- und Meßtechnik in der Produktion mikroelektronischer Bauelemente und Systeme
– bei technologischen Spezialausrüstungen und Produktionssystemen für Herstellung und Bestückung von Hybridbaugruppen und Leiterplatten mit dem Schwerpunkt bei SMT (Surface Mounted Technologie)
– bei Clean-Rooms und Clean-Technik sowie
– bei Grund- und Hilfsmaterialien der gesamten Zulieferbereiche Chemie, Metallurgie, Glas, Keramik und Maschinenbau sowohl für die Halbleitertechnik als auch für die Elektronik.

Die parallelen Fachtagungen „Halbleiterfertigung“ und „Maskentechnik für Mikroelektronik-Bausteine“ vermittelten einen Überblick über aktuelle technisch-technologische Probleme bei der Konstruktion und beim Einsatz von technologischen Spezialausrüstungen.

Am Beispiel des 1-MBit-DRAM, der den Stand der Technik darstellt, wurden aktuelle Probleme bei der Durchführung der Massenproduktion und deren Lösung aufgezeigt. So wurden insbesondere das unzureichende Niveau der Zuverlässigkeit und Verfügbarkeit der technologischen Spezialausrüstungen im Fertigungsbetrieb bemängelt und die verheerenden Folgen dieser nocht immer zu niedrigen Verfügbarkeit auf die Ausbeuten am Beispiel deutlich gemacht; die Lieferfähigkeit wurde beeinträchtigt, Marktanteile gingen verloren, und letztlich entstanden Schäden in Millionenhöhe für die IC-Hersteller.

Für die Technologien und VLSI-Bauelemente der 90er Jahre werden die Anforderungen an die technologischen Spezialausrüstungen und an die Halbleitergrund- und Hilfsmaterialien diskutiert.

Das Tempo der Entwicklung des Integrationsgrades von VLSI-Bauelementen hält unvermindert an und erzwingt die stetige Verkleinerung der minimalen Strukturabmessungen und die Einführung neuer Technologie-Generationen.

Die Technologieentwicklung in diesem Zeitraum wird von den dynamischen Speicherbauelementen, den 4-MBit-, 16-MBit- und 64-MBit-DRAMs repräsentiert, die als „Technologie-Lokomotiven“ fungieren (siehe Tafel).

DRAM	*minimale Strukturabmessung*	*Integrationsgrad [Transistoren/mm²]*	*Einführung in die Produktion*
64 KBit	*2 µm*	*4000 ... 8000*	*1982*
256 KBit	*1,5 µm*	*12000 ... 20000*	*1984/85*
1 MBit	*1 µm*	*30000 ... 50000*	*1987*
4 MBit	*0,8 µm*	*120000*	*1989*
16 MBit	*0,5 µm*	*400000*	*1991/92*

Chipherstellung (Zyklus I)

Für die Strukturübertragung durch Fotolithographie bei 1-MBit-DRAMs haben sich die lichtoptischen Waferstepper durchgesetzt. Um die für die Abbildung der Minimalstrukturen erforderliche Auflösung zu erreichen, wird gegenwärtig mit einer Lichtwellenlänge von 486 nm gearbeitet. Die Weiterentwicklung geht zur Lichtwellenlänge bis 248 nm (tiefes UV), das von Excimer-Lasern erzeugt wird. Parallel dazu wird der Einsatz von Röntgenstrahlen für die Lithographie erwogen, bei der eine praktische Auflösung bis in den Bereich von 0,2 ... 0,3 µm möglich wird. Der erste Prototyp eines Röntgensteppers für den Produktionseinsatz wurde auf der Messe vorgestellt und im Vortragsprogramm diskutiert. Durch die oben skizzierten Fortschritte bei der lichtoptischen Lithographie ist der Produktionseinsatz der Röntgenlithographie jedoch noch nicht entschieden.

Das brisanteste Problem der Chipfertigung beim 1-MBit-DRAM und den Folgegenerationen, dem 4-MBit- und 16-MBit-DRAM, sind die erneut erhöhten Anforderungen an die defektfreie Bearbeitung der Siliziumscheiben. Neue Clean-Rooms mit Staubklassen von 10 und besser, die einen kontrollierten Staubpegel für Partikel bis kleiner 0,1 µm sichern, erhöhen den Investaufwand und die Betriebskosten von Halbleiterfertigungslinien um den Faktor 1,5 gegenüber dem jeweils vorangegangenen Niveau. Man glaubt, daß diese Kostenexplosion für die Bauelementegenerationen ab dem 4-MBit-DRAM nur noch mit völlig neuen Konzepten der Clean-Rooms in wirtschaftlich vertretbaren Grenzen gehalten werden kann. Als favorisierter Lösungsansatz werden sog. „lokale Reinräume“ diskutiert, in denen die Maschinen durch eine klimatisierte Umhausung – „Clean- im Clean-Room“ – gestaltet wird. Siliziumscheiben werden dann nicht mehr durch den Menschen und nicht mehr offen im Cleanroom manipuliert oder bearbeitet. Die Bestückung der Maschinen erfolgt dann ausschließlich durch spezielles Handling bzw. Roboter. Hier hat sich als einheitliches System das sog. „SMIF“ (Standard Medanical Interface) favorisiert. Erste Beispiellösungen wurden an Messeständen der Fa. Meißner & Wurst und der Fa. Leitz gezeigt.

Ausgehend von dem oben Gesagten und unter Berücksichtigung der Erfahrungen, daß durch die technologischen Spezialausrüstungen während der Transport- und Manipuliervorgänge 25 % und bei der Durchführung der technologischen Bearbeitung der Siliziumscheibe verfahrensbedingt weitere 25 % der nachweisbaren Defekte erzeugt werden, müssen bei der Neukonstruktion völlig neue Wirkprinzipien durchgesetzt werden. So wird ausschließlich eine Einscheibenbearbeitung erfolgen; bewegte, partikelerzeugende Teile oberhalb der Scheibe wird es nicht mehr geben, neue Werkstoffe sowohl für Bewegungselemente als auch für die Bearbeitungskammern und Reak-

toren werden entwickelt. Insgesamt wird die Gesamtkonstruktion und die Ausführung der Oberflächen der TSA cleanroomgerecht und der Staubklasse 1 angepaßt sein. Ebenso werden solche neuen Verfahrensschritte entwickelt, die einen kontaminationsfreien Prozeßablauf gestatten; das betrifft besonders das Plasmaätzen, das Sputtern und die Schichtabscheidung aus der Dampfphase (CVD-Prozesse). Nicht zuletzt wird in Umsetzung neuester Erfahrungen beim Plasmaätzen von Aluminiumschichten und bei CVD-Prozessen an Verfahren gearbeitet, die entweder ohne toxische Ätz- oder Reaktionsgase auskommen oder die eine schadlose Entsorgung ermöglichen.

Japan und die USA dominieren sowohl in der Produktion und dem weltweiten Umsatz mikroelektronischer Bauelemente mit ca. 80 % und bei technologischen Spezialausrüstungen mit ca. 90 % Anteil. Als Folge daraus ergibt sich z. B. für die BRD, daß ihr Bedarf an Halbleiterbauelementen nur zu 40 % und bei technologischen Spezialausrüstungen nur zu 10 ... 20 % aus Eigenaufkommen gedeckt wird.

Förderprogramme der europäischen Regierungen gegenüber großen Konzernen zur Verringerung des Abstandes zum internationalen Spitzenniveau zeitigen erste Erfolge. So wurden 1987 die Produktion des 1-MBit-DRAM aufgenommen und erste Muster des 4-MBit-DRAM und 1-MBit-SRAM vorgestellt. Europäische TSA-Hersteller partizipieren an den Investaufwendungen für dieses „Mega-Programm". Durch eigene Entwicklungen von TSA und der Prozeßmeßtechnik für den Zyklus I entstanden neue durchaus konkurrenzfähige Erzeugnisse wie Beschichter und Entwickler der Fa. Convac, lichtoptische Stepper der Fa. ASM, Röntgenstepper der Fa. Karl Süß, Plasmaätzer der Firmen Alcatel Electrotech und Laybold, Sputteranlagen der Firmen Balzers und Heraeus, CVD-Anlagen der Firmen ASM, PLASMOS, Controtherm und LPE, Prozeß und Kontrolltechnik von Heidelberg Instruments, Leitz sowie Chemie- und Naßarbeitsplätze.

In den Konstruktionsbüros der führenden europäischen TSA-Hersteller wird in enger Zusammenarbeit mit Bauelementeherstellern und wissenschaftlichen Einrichtungen eine neue Ausrüstungsgeneration für den 4- bzw. 16-MBit-DRAM vorbereitet, die wie folgt zu charakterisieren ist:

- 150 bis 200 mm Scheibendurchmesser
- kleinste realisierte Strukturbreiten auf dem Schaltkreis 0,7 ... 0,5 μm
- Einsatz des lichtoptischen Steppers zur Strukturübertragung
- ausschließlicher Einsatz von Plasmaätzern mit neuen Wirkprinzipien (RIE- und Mikrowellenätzen, Mehrkammerätzer mit sehr gutem, kontrolliertem Vakuum und kontrollierten Prozeßabläufen), die sowohl eine gute Selektivität als auch Hochrateätzen der Strukturen gestatten
- weiteres Verdrängen der plasmageätzten Prozesse zur Abscheidung von Schichten
- Einscheibenprozesse
- mikroprozessorgesteuerte und mikroprozessorüberwachte, durch Roboter bestückte TSA, die einen wesentlich größeren Anteil an Insito-Messungen aufweisen
- flexibel automatisierte Inseln innerhalb des Gesamtfertigungsablaufes in den Scheibenprozessen und
- Rechnerüberwachung und -steuerung des Gesamtprozesses (informatorische Verkopplung).

Bauelemente-Montage (Zyklus II)

Im Zyklus II konzentrieren sich TSA-Hersteller auf die Anpassung vorhandener TSA auf SMD-Gehäuse. Bei ausgewählten TSA wird nach neuen Lösungen gesucht, so für das Kennzeichnen von Bauelementen, wobei noch nicht entschieden ist, ob sich bei SMD-Bauelementen der herkömmliche Offset-Druck oder das neue flexiblere, aber teurere Laserkennzeichnen durchsetzen wird. Erstmalig werden nach erfolglosen Versuchen der letzten zwei Jahre Neukonstruktionen von sog. Multiplunger-Verkappungsmaschinen vorgestellt. Mit den Multiplunger-Verkappungsmaschinen in Verbindung mit Neukonstruktionen von Automaten für das Beschneiden und Abbiegen der Bauelementeanschlüsse stehen für den Zyklus II nunmehr durchgängig Automaten für die kontinuierliche Bearbeitung von Bauelementen im Trägerstreifenverband zur Verfügung, die von Magazin zu Magazin arbeiten. Damit wird es möglich, flexibel verkettete automatische Linien für die Montage von mikroelektronischen Bauelementen in herkömmlichen DIL- oder in SMD-Gehäusen aufzubauen. Diese Aufgabe verbleibt aber beim Bauelementehersteller; Anbieter für durchgängig automatisierte Linien gibt es nicht.

Mit der Weiterentwicklung von oberflächenmontierbaren Bauelementen zu sehr hohen Anschlußzahlen über 140 hinaus wird sich ein neues Gehäuse, das Trägerfilm-Carrier, durchsetzen. Für die Montage solcher Trägerfilm-Carriers steht mit der TAB-Technologie (Tape Automated Bonding) eine durchgängig automatisierbare Montagetechnologie hoher Flexibilität und Entwicklungsfähigkeit zur Verfügung. Zunehmende Aktivitäten der Bauelemente- und TSA-Hersteller sowie der Zulieferindustrie (Trägerfilme aus Polyimid), weisen darauf hin, daß TAB das Montagesystem für VLSI und Gate Arrays für die 90er Jahre sein wird.

Hybrid-Montage

Trotz fortschreitender Integration von Systemlösungen in Form von ASICs (anwendungsspezifische Schaltkreise) und der Erhöhung der Pakkungsdichte auf der Leiterplatte durch SMT wächst die Produktion von Hybridbaugruppen in Europa weiter mit Steigerungsraten von mehr als 25 %. Die technologischen Grenzen zwischen der Hybridmontage und der SMD-Bestückung von Leiterplatten verwischen sich zwar zunehmend; die Hybrid-Baugruppe mit Keramik-Bauelementeträger und Mehrebenenverdrahtung wird für Spezialanwendungen in schnellen Rechnern und für Baugruppen mit hohen Zuverlässigkeitsanforderungen (Weltraum, Militärtechnik) ihre Bedeutung behalten.

Die Weiterentwicklung in der Hybridtechnik geht zu Trägermaterialien hoher Wärmeleitfähigkeit, zu Leitbahnabmessungen kleiner 50 μm und zur Nacktchipverarbeitung. Für die letztgenannte Aufgabe stellen Bonderfirmen nunmehr auch automatische Chip- und Drahtbonder zur Verfügung.

Herstellung elektronischer Baugruppen

Flächenmäßig und nach Zahl der Hersteller hatten die Fertigungsanlagen, Materialien und Zubehör für Fertigung elektronischer Baugruppen auf Leiterplatten den größten Anteil auf der Productronica. Mit einem weltweiten Umsatz von ca. 12 Mill. US$ erreichen Leiterplatten fast die Hälfte des weltweiten Umsatzes von integrierten Schaltkreisen, damit wird ihr ökonomisches Gewicht in der Elektronikproduktion deutlich.

Entsprechend dem weitaus größeren Anteil westeuropäischer Firmen bei der Anwendung der Mikroelektronik im Vergleich zur Bauelementeherstellung ist der anteilige Verbrauch Europas bei Leiterplatten und entsprechend die TSA-Produktion für die Leiterplattenherstellung gewichtiger als in der Halbleiterproduktion.

Die Productronica '87 stand im Zeichen der sich durchsetzenden Oberflächenmontage-Technologie.

Das Angebot bei TSA reicht von Handbestückungsplätzen mit max. 1000 BE/h bis zu Bestückungsautomaten mit 4–10 TBE/h. Die Bauelemente werden in allen Lieferformen, vom Schüttgut über Stangenmagazine, Blistergurtung bis hin zu BE-Paletten und als Trägerfilm-Carriers (TAB) von laufender Rolle verarbeitet.

Bei der Mehrzahl der Bestückungsanlagen mit Einzelbestückung wird eine Positioniergenauigkeit von ± 0,1 mm, bei Anwendung von Revolverköpfen eine Genauigkeit von ± 0,2 mm erreicht. Für die neuesten Automaten werden optische Systeme der Bildverarbeitung als Bestükkungshilfen eingesetzt.

Für das Löten der Baugruppen dominieren z. Z. das Doppelwellen- und das Infrarotlöten; das Dampfphasenlöten wird sich aus Gründen des Gesundheitsschutzes und der Zuverlässigkeit nicht durchsetzen. Das Laserlöten wird sich Teilgebiete, z. B. das Löten von thermisch sensiblen Bauelementen, erobern.

Trotz der enormen materiell-ökonomischen Vorteile von SMT werden derzeit in Europa nur 10 % aller elektronischen Baugruppen in reiner SMT gefertigt, 40 % der Leiterplatten werden mischbestückt. Auch für 1990 rechnet man noch bei 50 ... 60 % aller Leiterplatten mit Mischbestükkung.

Ausgehend von den sich schnell verändernden Anforderungen des Marktes ist aus Konkurrenzgründen eine flexible Reaktion in der Baugruppenfertigung durch auftragsgebundene Produktion mit kurzen Durchlauf- und Lieferzeiten erforderlich.

Diese Anforderungen können nur von flexiblen Fertigungssystemen erfüllt werden.

Im Gegensatz zur Halbleitertechnik werden solche Fertigungssysteme für die Baugruppenmontage von großen TSA-Herstellern angeboten, die in der Regel modular aufgebaut sind und damit den verschiedenen Anforderungen leicht angepaßt werden können. Je nach geforderten Produktionsvolumen sind Linien mit unterschiedlichem Automatisierungsgrad realisierbar:

- Off-line-Systeme ohne mechanische und ohne informatorische Verkettung der TSA; der Leiterplattentransport erfolgt von Hand.
- In-line-Systeme mit mechanischer Verkettung der TSA über ein starres Transportsystem und Zwangsführung des Leiterplattentransports. Es besteht die Notwendigkeit von Leiterplatten-Puffern, um bei Störungen einzelner TSA nicht die ganze Linie stoppen zu müssen.
- On-Line-Systeme mit Ankopplung der TSA der Linie an ein zentrales Transportsystem mit rechnergestützter Zuführung der Leiterplatten zu jeder TSA. Dadurch wird hohe Flexibilität und Systemverfügbarkeit erreicht.

Dr. Werner Prischmann

PC-Einsatz in Gießereien

Der BFA „Produktionsorganisation" der KDT Magdeburg läßt es zu einer Tradition werden, daß alljährlich eine Veranstaltung für Fachkollegen zur Produktionsplanung und -steuerung (PPS) aus Gießereien der DDR durchgeführt wird. So fand am 26. und 27. November 1987 das zweite Gießereiseminar zum Thema „PC-Einsatz in Gießereien" statt. Die Teilnehmerzahl von 62 Fachkolleginnen und -kollegen macht deutlich, daß die Thematik ein breites Echo findet. Das Ziel dieses Seminars bestand darin, Voraussetzungen für den PC-Einsatz abzuleiten, Anwendungsbedingungen und -möglichkeiten aufzuzeigen, praktische Einsatzfälle zu demonstrieren und den Erfahrungsaustausch bei der Einsatzvorbereitung und praktischen Nutzung dieser Technik vorzunehmen. Diesem Anliegen wurden die einzelnen Beiträge gerecht.

Prof. Dr. sc. techn. Gottschalk (TUM) informierte in seinem Einführungsvortrag darüber, welche Schritte bei der weiteren Mechanisierung und Automatisierung der Produktionsprozesse zur Automatisierung des Informationsflusses in Verbindung mit dem Rechnereinsatz zu gehen sind und welche Konsequenzen sich daraus für die PPS ableiten.

Voraussetzung für eine fundierte Produktionsplanung und -steuerung ist die technologische Produktionsvorbereitung. Die Vielzahl der Sortimente und unterschiedlichen Technologien fordert rationelle Methoden. Dazu stellte *Dr. Meister* (VEB Baukema Leipzig) ein umfangreiches Konzept mit Bausteinlösungen für den PC 1715 vor und demonstrierte bereits praktisch genutzte Teile.

Von der TUM, WB Betriebsgestaltung (Leiter *Prof. Gottschalk*) wurden auf der Basis von BASIC Dateikonzepte zur PPS in Gießereien erarbeitet, die einen minimalen Speicherplatzbedarf und minimale Zugriffszeiten aufweisen sowie einen schrittweisen Ausbau ermöglichen. Diese Lösungen und ihre praktische Umsetzung für ausgewählte Anwendungsfälle wurden in den Vorträgen von *Glistau* (TUM)/*Stark* (VEB Gießerei Ueckermünde), *Bärecke* (TUM)/*Köllmer* (VEB MEGU Wernigerode) und *Leister* (TUM)/*Wandtke* (VEB APAG Potsdam) aufgezeigt. Dabei standen Probleme des Aufbaus der Stammdatei, Aufbau und Nutzung von Dateien und Recherche- bzw. Abrechnungsroutinen für Fortschrittskontrolle sowie die Rationalisierung des

Fortsetzung auf Seite 189

Direkte Verarbeitung von dBASE/REDABAS-Dateien mit PASCAL

Es wurde ein Konzept und eine Lösungsmethode der direkten Verarbeitung von dBASE/REDABAS-Dateien mit TURBO-PASCAL (bzw. PASCAL 880/S) entwickelt. Gegenüber anderen Methoden wird dabei auf einen Konvertierungslauf verzichtet und damit keine zusätzliche Kapazität auf dem externen Speichermedium benötigt. Die dBASE/REDABAS-Dateien bleiben unverändert, so daß sie jederzeit wieder mit dBASE/REDABAS verarbeitet werden können. Die geschaffene Lösung gestattet sowohl das Schreiben, als auch das Lesen von Datensätzen.
Die entsprechenden Programmteile liegen in Prozedur- oder Funktionsform vor, so daß sie problemlos angewendet werden können. Die Lösung ist mit ausführlichen Beispielen auch für PASCAL-Einsteiger hinreichend dokumentiert.
Durch die Anwendung dieses Konzeptes können die hervorragenden Möglichkeiten von dBASE/REDABAS bei der Datenerfassung und -verwaltung mit der schnellen Verarbeitung und Auswertung in PASCAL vereint werden. Das bringt besonders bei mathematischen Berechnungen, Auswertungen umfangreicher Datenbestände und bei der parallelen Verarbeitung von mehreren Dateien große Laufzeitvorteile.
Die vorgestellte Lösung ist lauffähig auf 8- und 16-Bit-Technik unter dem Betriebssystem SCP.

INTERFLUG, Betrieb Verkehrsflug, Betriebsteil Flugtechnik, Berlin-Schönefeld, 1189; Tel. 6 72 74 12

Scholz

Wörterbuch

Der hohe Aufwand an Schreibarbeiten bei der Schaffung von Wörterbüchern wird durch den Einsatz eines PC mit Hilfe von REDABAS (dBASE II) stark reduziert. Die Lexik beliebiger Texte kann für ein- oder zweisprachige Wörterbücher in Sprachen mit lateinischen wie auch kyrillischen Schriftzeichen erfaßt, nach gewünschten Kriterien charakterisiert und geordnet werden.
Linguistisch-statistische Auswertungen sind möglich.
Wir bieten die dazu notwendige Software an. Konsultationen zur Übermittlung unserer Erfahrungen sind möglich.

Agraringenieurschule Stadtroda, Lehrgebiet Russisch, Stadtroda, 6540; Tel. 22 102

Dr. B. Fisch

Hardwarelösung für PC-Kompatibilitätsprobleme

Zum Austausch von Nutzerdaten mittels Diskette für die Mehrfachbenutzung in verschiedenen Anwendungsgebieten wurde eine Hardwarelösung entwickelt und realisiert, die sich seit ihrer Übergabe zur Nutzung vielfach und hervorragend bewährt hat.
An einem Bürocomputer A 5120 sind physisch mehrere Diskettenlaufwerke (Anzahl der Laufwerke > 4) beliebiger Typen angeschlossen. Aus dieser Laufwerk-Vielfalt können durch einfachste Bedienung vier Laufwerke logisch ausgewählt werden.
Dabei besteht zusätzlich die Möglichkeit, einer beliebigen Kombination der Laufwerk-Reihenfolge, d. h., jeder Laufwerk-Typ kann z. B. als Laufwerk A oder auch als systemlogisches Laufwerk benutzt werden. Dies ist für die Benutzung mancher Softwarekomponenten und zur Überwindung von Besonderheiten mancher CP/M-kompatibler Betriebssysteme von Bedeutung.
Durch eine entsprechende Generierung verschiedener Betriebssystemvarianten zum SCPX 1526 (z. B. für die Benutzung der Laufwerke 6400 mit einfacher bzw. doppelter Dichte) ergeben sich weitere Kombinationsmöglichkeiten.
Mit dieser Lösung konnten bisher alle Datenaustausche von allen nach allen Diskettentypen in fast allen Formaten realisiert werden, dabei Datenerfassungsaufwendung erheblich eingespart und die Benutzung gleicher Datenbestände mittels verschiedener Gerätetechnik ermöglicht werden.
Die Erfahrung hat gezeigt, daß es vorteilhaft und ausreichend ist, in entsprechenden Struktureinheiten über eine solche Gerätevariante zu verfügen; z. B. ein Gerät im Betrieb oder je Betrieb eines Kombinates ein Gerät, wenn sie territorial ungünstige Standpunkte aufweisen.

VEB Bau- und Montagekombinat Erfurt, Kombinatsleitung, Bereich DODV, Abt. GV, Koll. Liebsch, Juri-Gagarin-Ring 152, Erfurt, 5010

Liebsch/Baumgärtl

Reassemblierung

In unserem Bereich wurde ein Programm zur Reassemblierung von Maschinenkode der Einchipmikrorechner U881/U882 entwickelt. Das Programm ist in PASCAL-T (TURBO-PASCAL) geschrieben und läuft unter dem Betriebssystem SCP (CP/M). Es kann auch auf Rechner mit dem Betriebssystem MS-DOS 3.0 implementiert werden. Das Programm besitzt folgende Eigenschaften:
- Kommandos zur Steuerung der Reassemblierung, wie Eingabe von Tabellenbereichen
- Ausgabe des reassemblierten Files auf dem Bildschirm und auf Diskette
- Reassemblierung von maximal 8 KByte Maschinenkode.

Die Ausgabe der erzeugten Quelldatei erfolgt in der Mnemonik, wie sie beim Programmsystem PLZ/ASM unter dem Betriebssystem UDOS verwendet wird.

Humboldt-Universität zu Berlin, Sektion Elektronik, Bereich 5, Invalidenstr. 110, 1040 Berlin

Dr. F. Winkler

Programmsystem comFORTH Version 1.10

Das seit 1985 zur Nachnutzung angebotene Programmsystem comFORTH der Wilhelm-Pieck-Universität Rostock wurde im vergangenen Jahr stark überarbeitet. Mit der Version 1.10 steht ein System zur Verfügung, das qualitativ neue Leistungsmerkmale aufweist. Dazu gehört u. a. eine verbesserte Wörterbuchverwaltung, die das komprimierte Laden von Erweiterungen gestattet.
Die Version 1.10 arbeitet auf SCP-kompatiblen Betriebssystemen mit den Prozessoren Z80 und i8086/8088. Damit ist die Nutzung bisheriger comFORTH-Software bei voller Kompatibilität auf der Hochwortebene auch auf dem A 7100 möglich.
Die als Grundausstattung angebotenen Komponenten Editor, Assembler und Debugger wurden vollständig überarbeitet und bieten erhöhten Bedienkomfort. Das Angebot an Erweiterungspaketen wurde um dokumentationsunterstützende Komponenten bereichert.
Die Programmierung spezieller Rechnerkonfigurationen ist mit Hilfe von Crosscompilern einer neuen Generation für die Prozessoren i8086/8088, Z80 und Z8 möglich. Die prozessorspezifische Kernschicht wird in Form von Quellenbibliotheken bereitgestellt. Durch die Crosscompiler wird auch die Metacompilierung interaktiver FORTH-Systeme unterstützt.
Die Quellen der comFORTH-Kernsysteme gehören zum Lieferumfang.
Interessenten an einem ausführlichen Informationsblatt wenden sich bitte nur schriftlich an die
Wilhelm-Pieck-Universität Rostock, Sektion Technische Elektronik, Bereich Automatische Steuerungen, Albert-Einstein-Straße 2, Rostock 6, 2500

Prof. Dr. Hormann

CTRL-Taste für K 7637

Nicht alle Modifikationen der seriellen Tastatur K 7637 sind mit einer CTRL-Tastatur ausgestattet. Um auch diese Modelle unter SCP zu nutzen, wurde eine Hardwareerweiterung entwickelt, mit der eine beliebige Taste die CTRL-Funktionen erfüllt.
Die kleine Leiterkarte findet im Tastaturgehäuse Platz.
Instandhaltungswerk Sicherungs-, Fernmelde- u. Prozeßautomatisierungstechnik, Bahnhofsvorplatz Güstrow, 2600

Moras

Programmpaket für Handwerksbetriebe

1. Schreiben von Handwerkerrechnungen
nach PA 211, PA 215 und PA 4511/12 (ÖVV)
für Dachklempnerarbeiten
Sanitärinstallation
Gasinstallation
Heizungsinstallation
erweiterbar auf andere Gewerke
2. Erstellen von Monats- und Jahresbilanzen für
Abt. Finanzen Preisstützung
HQ – Bogen und Statistik
Materialverbrauch durch Rechnung
Kundenkartei
3. Lohnabrechnung
mit Abspeicherung und Jahresabrechnung
einschl. Lohnsummensteuer
4. Materialkartei Einkauf und Inventur
mit Errechnung von Durchschnittspreisen und Druck der Jahresinventurlisten
5. Kassenbuch und Anlagenverzeichnis
6. Jahresabschluß mit Anrechnung der Daten aus 1. bis 5.
7. Serviceprogramm
Preisänderungen
Neuanlage der Dateien für neues Jahr usw.

Das Programmpaket ist lauffähig auf ATARI 130 XE mit einem Diskettenlaufwerk 1050 und Drucker mit Centronics-Interface (z. B. Robotron K6 311) und IBM-XT-kompatiblem PC (z. B. Schneider PC 1512) mit einem Laufwerk.

Alle betriebsspezifischen Daten befinden sich in Dateien, welche mittels des Serviceprogramms vom Nutzer leicht verändert werden können. Während der Abarbeitung der Programme erfolgen mit Ausnahme von 6. keine Diskettenzugriffe!

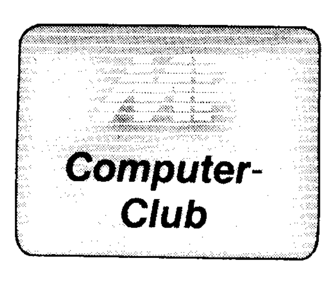
Dachklempnerei und Sanitärinstallation Roland Fuchs, Ernst-Thälmann-Str. 46, Prenzlau, 2130; Tel. 20 38

Fuchs

Fortsetzung von S. 188

Versandes im Vordergrund. Welche gerätetechnischen Möglichkeiten und Grenzen derzeit für die Betriebsdatenerfassung in Gießereien bestehen, wurden von *Enseleit* (TUM)/*Kolb* (ZIfV Berlin) erläutert, auch wurde die Kopplung mobiler Datenerfassungsgeräte mit PC-Technik demonstriert.
Mit diesen Maßnahmen und Lösungen, die einen PC-Einsatz in der Steuerebene erfordern, ergeben sich neue Möglichkeiten zur Produktionsplanung und Disposition sowie der Einbindung der Instandhaltungsorganisation. Dazu wurde eine Reihe von Lösungen von *Dr. Schenk/Wiemer* (TUM) und *Freiboth/Oertel* (TUM) vorgestellt.
Neben diesen Lösungen wurde von *Kühne/Kumminek* (VEB SKET Magdeburg) sowie *Bardubitzki* (VEB MEGU Finsterwalde) erläutert, welche Möglichkeiten bei der Nutzung von Standardsoftware (REDABAS) für die PPS bestehen und welche Erfahrungen bei der Einführung neuer Organisationslösungen unter Nutzung von PC gesammelt werden konnten.
Dr. Müller/Meyer (TUM)/*Baum* (VEB Baukema) erläuterten eine Methode zur rechnergestützten Projektierung von Formanlagen und wiesen dabei nach, wie stark bereits bei der technologischen Projektierung Probleme der Organisation beachtet werden müssen.
Bei den umfangreichen Rechnerdemonstrationen sowie den individuellen Vorführungen der Lösungen in den CAM-Laboren der TUM wurde der Erfahrungsaustausch fortgesetzt, was mit zum Gelingen dieser Veranstaltung beitrug.

Dr. Schenk



Bildschirmattribute beim A 7100

Auf immer mehr Arbeitsplätzen kann man den 16-Bit-Arbeitsplatz-Computer A 7100 von Robotron antreffen. Abgesehen von sehr deutlich werdenden Unzulänglichkeiten des Betriebssystems SCP 1700, die dem Nutzer vor allem beim Bildschirmlöschen und beim Bildschirmaufbau sowie bei Dateitransfer auffallen, beinhalten die mitgelieferten Programmiersysteme (BASIC, REDABAS) und unter SCP 1700 (CP/M 86) lauffähigen Softwareentwicklungssysteme (TURBO-PASCAL) keine Unterstützung zur komfortablen Bildschirmsteuerung. Das kann sich der Nutzer zwar in den umfangreichen, gedruckten Materialien, die Robotron zur Verfügung stellt, ansehen, besser wäre natürlich, die Software würde das von sich aus leisten.
Im folgenden werden deshalb für die Systeme SCP 1700-BASIC, REDABAS (dBASE II) und TURBO-PASCAL die notwendigsten Steuerfolgen als Beispiele dargestellt. Man kann sie beliebig kombinieren und anhand des Anleitungsmaterials vervollkommnen.

Dr. Knut Löschke

```
1. BASIC
-------------------
10 REM ******** BEISPIELPROGRAMM FUER BILDSCHIRMATTRIBUTE *******
11 REM          CLS$ löscht den Bildschirm  -------> CLS-Befehl
12 REM          FNLOCATE$(Z,S) lokalisiert den CURSOR bzw. die
13 REM                          Ausgabe auf dem Bildschirm mit
14 REM                          Z=Zeile,S=Spalte  --> LOCATE-Befehl
15 REM          N$   Attribut normale Heligkeit
16 REM          H$            doppelte Helligkeit
17 REM          NI$           normal invers
18 REM          HI$           hell invers
19 REM          NU$           normal unterstrichen
20 REM          HU$           hell unterstrichen
21 REM          NB$           normal blinkend
22 REM          HB$           hell blinkend
50 REM ***************************************************
51 REM           Definitionen
55 ESC$=CHR$(27)
60 CLS$=ESC$+"[2J"
70 N$=ESC$+"[Om"
80 H$=N$+ESC$+"[1m"
90 NI$=N$+ESC$+"[7m"
100 HI$=H$+ESC$+"[7m"
110 NB$=N$+ESC$+"[5m"
120 HB$=H$+ESC$+"[5m"
130 NU$=N$+ESC$+"[4m"
140 HU$=H$+ESC$+"[4M"
150 DEF FNLOCATE$(Z,S)=ESC$+"["+MID$(STR$(Z),2)+";"
                  +MID$(STR$(S),2)+"H"
152 REM          Beispiel der Anwendung
160 PRINT CLS$
180 PRINT FNLOCATE$(1,1)+N$+"NORMAL"
200 PRINT FNLOCATE$(2,2)+H$+"HELL"
220 PRINT FNLOCATE$(3,3)+NI$+"NORMAL-INVERS"
240 PRINT FNLOCATE$(4,4)+HI$+"HELL-INVERS"
260 PRINT FNLOCATE$(5,5)+NU$+"NORMAL-UNTERSTRICHEN"
280 PRINT FNLOCATE$(6,6)+HU$+"HELL-UNTERSTRICHEN"
300 PRINT FNLOCATE$(7,7)+NB$+"NORMAL-BLINKEND"
320 PRINT FNLOCATE$(8,8)+HB$+"HELL-BLINKEND"
340 PRINT FNLOCATE$(10,10)+N$+"PROGRAMM NORMAL BEENDET"
350 END
```

```
2. REDABAS (DBASE-II)
---------------------
*      Definitionen der Attribute analog BASIC
*
SET TALK OFF
ERASE
STORE CHR(27) TO ESC
STORE ESC+"[Om" TO N
STORE N+ESC+"[1m" TO H
STORE N+ESC+"[7m" TO NI
STORE H+ESC+"[7m" TO HI
STORE H+ESC+"[4m" TO HU
STORE N+ESC+"[4m" TO NU
STORE N+ESC+"[5m" TO NB
STORE H+ESC+"[5m" TO HB
*      Beispiele der Anwendung
@ 1,1 SAY N+"NORMAL"
@ 2,2 SAY H+"HELL"
@ 3,3 SAY NI+"NORMAL-INVERS"
@ 4,4 SAY HI+"HELL-INVERS"
@ 5,5 SAY NU+"NORMAL-UNTERSTRICHEN"
@ 6,6 SAY HU+"HELL-UNTERSTRICHEN"
@ 7,7 SAY NB+"NORMAL-BLINKEND"
@ 8.8 SAY HB+"HELL-BLINKEND"
@ 10,10 SAY N+"PROGRAMM NORMAL BEENDET"
```

```
3. Turbo-PASCAL
-------------------
PROGRAM ATTRIB;
( Definitionen der Attribute analog BASIC )
VAR ESC : STRING[1]     ;
    N   : STRING[4]     ;
    H   : STRING[8]     ;
    NI  : STRING[8]     ;
    HI  : STRING[12]    ;
    NB  : STRING[8]     ;
    HB  : STRING[12]    ;
    NU  : STRING[8]     ;
    HU  : STRING[12]    ;
BEGIN
    ESC:=CHR(27)        ;
    N   :=ESC+'[Om'     ;
    H   :=N+ESC+'[1m'    ;
    NI  :=N+ESC+'[7m'    ;
    HI  :=H+ESC+'[7m'    ;
    NB  :=N+ESC+'[5m'    ;
    HB  :=H+ESC+'[5m'    ;
    NU  :=N+ESC+'[4m'    ;
    HU  :=H+ESC+'[4m'    ;
CLRSCR;
GOTOXY(1,1);
WRITELN(H+'PROGRAMM > ATTRIB < BEISPIEL FUER BILDSCHIRMATTRIBUTE');
GOTOXY(3,3);
WRITELN(N+'NORMAL');
GOTOXY(4,4);
WRITELN(H+'HELL');
GOTOXY(5,5);
WRITELN(NI+'NORMAL-INVERS');
GOTOXY(6,6);
WRITELN(HI+'HELL-INVERS');
GOTOXY(7,7);
WRITELN(NU+'NORMAL-UNTERSTRICHEN');
GOTOXY(8,8);
WRITELN(HU+'HELL-UNTERSTRICHEN');
GOTOXY(9,9);
WRITELN(NB+'NORMAL-BLINKEND');
GOTOXY(10,10);
WRITELN(HB+'HELL-BLINKEND');
GOTOXY(12,12);
WRITELN(N+'PROGRAMM NORMAL BEENDET');
END.
```

Stringarithmetik für BASIC-Programme

Der Basicinterpreter der Kleincomputer arbeitet mit einer Zahlengenauigkeit von 6 Stellen. Dies reicht für wissenschaftliche Berechnungen im allgemeinen Anwendungsbereich des KC völlig aus. Sollen jedoch ökonomische Aufgabenstellungen realisiert werden, ist diese Genauigkeit unzureichend, da bereits im Wertebereich ab 10000 Mark die Pfennige „verschwinden". Um den KC 85/3 auch im Bereich der Ökonomie einsetzen zu können, ohne den Hauptbuchhalter dem Herzinfarkt nahe zu bringen, wurde die folgende Stringarithmetik entwickelt.

Arbeitsprinzip

Jede zu verarbeitende Zahl, deren Eingabe als String erfolgen muß, wird im Durchlauf durch das Unterprogramm **Zerlegung** in einen höherwertigen und einen niederwertigen Teil aufgespaltet. Beide Teile werden in getrennten Variablen gespeichert. Es hat sich hier als günstig erwiesen, Arrays zur Speicherung zu verwenden, da durch die Indizierung der Feldvariablen eine einheitliche Zuordnung zu den beiden Teilen der Zahl erfolgen kann.

Beispiel:

```
DIM X (1, x)
       |  └ laufende Nummer.
       |
       |    1 = niederwertiger Teil
       └─── 0 = höherwertiger Teil
```

Für die Ausführung von Rechenoperationen werden beide Zahlenteile getrennt bearbeitet und anschließend Übertragungsoperationen ausgeführt. Im Demonstrationsbeispiel ist durch entsprechende Kommentierung die prinzipielle Verfahrensweise erkennbar.
Die Aufspaltung der Zahl erfolgt zwischen 2. und 3. Stelle links des Kommas. Es wird immer auf 2 Dezimalstellen gerundet. Dadurch ist der Zahlenbereich von 0 bis 99999999.99 darstellbar.

Wird für den Druck ohne die Darstellung auf dem Bildschirm eine Zahl benötigt, werden beide gespeicherten Teile im Unterprogramm **Zusammensetzen** wieder vereint und als String dargestellt. Gleichzeitig mit dem Zusammensetzen erfolgt eine Formatierung des Strings auf eine konstante Länge, so daß ohne Schwierigkeiten „kommabündig" geschrieben werden kann.

Beschreibung der Programmkomponenten

1. Zerlegung: Die zu bearbeitende Zahl wird als A○ an das UP übergeben. Es sind 8 Stellen vor dem Komma zulässig. Der Programmaufruf erfolgt durch GOSUB 100.
Die Rückgabe vom UP erfolgt in den Variablen:
A1 = HT und A2 = NT.
2. Übertrag Addition: Nach Durchführung einer Addition, wobei die höherwertigen und die niederwertigen Teile beider Summanden getrennt addiert werden, erfolgt die Übernahme und Rückgabe in den Variablen A1=HT und A2=NT (GOSUB 320).
3. Zusammensetzen: Die beiden Zahlenteile werden an A1 und A2 übergeben und stehen nach UP-Aufruf (GOSUB 370) als A○ in formatierter Form, mit 11 Zeichen Länge/2 Dezimalstellen, für eine Darstellung zur Verfügung.
4. Multiplikation: Bei der Multiplikation darf nur ein Faktor eine zerlegte Zahl sein. Der zweite Faktor muß einen positiven numerischen Wert besitzen, der in einer numerischen Variablen angespeichert werden kann.
Variante 1:
Der 1. Faktor wird im UP **Multiplikation** entweder als A○ übergeben (dann Aufruf mit GOSUB 610) oder, wenn die Zahl bereits zerlegt vorliegt, als A1 und A2 (dann Aufruf mit GOSUB 620).
Der 2. Faktor wird immer als Variable A6 übergeben.
Die Rückgabe des Multiplikationsproduktes erfolgt als A○.
Variante 2:
Die Multiplikation wird unmittelbar im BASIC-Programm mit der bereits zerlegten Zahl und dem 2. Faktor ausge-

```
5 GOTO 1000
10 REM. VERWENDETE VARIABLE
20 REM A1/A2/A3/A4/A5/A6/A7/A8/A9/A0
30 REM AA/AB/AC/AD/AE/AF/AG/AH
40 REM A1¤/A2¤/A3¤/A4¤
50 REM *** STRINGARITHMETIK ***
60 REM ++++++++++++++++++++++++++++++++++++++++
70 REM UP *ZERLEGUNG*
80 REM MAX FORMAT 8 STELLEN VOR KOMMA
90 REM UEBERGABE ALS A¤
100 AA=LEN(A¤)
110 IF AA=0 THEN A¤="        0.00"
120 A5=INSTR(".",A¤)
130 IF A5=0 THEN A¤=A¤+".00":GOTO 220
140 IF A5=AA-1 THEN A¤=A¤+"0":GOTO 220
150 IF A5=AA THEN A¤=A¤+"00":GOTO 220
160 IF A5<AA-2 THEN 170:ELSE 220
170 AB=VAL(RIGHT¤(A¤,1)):A¤=LEFT¤(A¤,AA-1)
180 A7=VAL(RIGHT¤(A¤,1)):IF AB>=5 THEN A7=A7+1
190 A3¤=STR¤(A7):A3¤=RIGHT¤(A3¤,1)
200 A¤=LEFT¤(A¤,AA-2):A¤=A¤+A3¤
210 GOTO 100
220 AA=LEN(A¤)
230 A¤=STRING¤(11-AA," ")+A¤
240 A¤=LEFT¤(A¤,8)+RIGHT¤(A¤,2)
250 A1=VAL(LEFT¤(A¤,6)):A2=VAL(RIGHT¤(A¤,4))
260 REM RUECKGABE
270 REM A1=HOEHERW. TEIL    A2=NIEDERW. TEIL
280 RETURN
290 REM ++++++++++++++++++++++++++++++++++++++++
300 REM *** UEBERTRAG ADDITION ***
310 REM UEBERGABE UND RUECKGABE ALS A1/A2
320 A3=INT(A2/10000)*10000:A2=A2-A3
330 A1=A1+A3/10000:RETURN
340 REM ++++++++++++++++++++++++++++++++++++++++
350 REM *** ZUSAMMENSETZEN ***
360 REM UEBERGABE ALS A1/A2
370 IF A1=0 THEN A¤="      ":GOTO 410
380 A¤=STR¤(A1):AA=LEN(A¤)
390 A¤=RIGHT¤(A¤,(ABS(AA-1))):AA=LEN(A¤)
400 A¤=STRING¤((6-A)," ")+A¤
410 IF A2=0 THEN A4¤="00.00":GOTO 460
420 A4¤=STR¤(A2):A5=LEN(A4¤)
430 A4¤=RIGHT¤(A4¤,A5-1)
440 A4¤=STRING¤(4-LEN(A4¤),"0")+A4¤
450 A4¤=LEFT¤(A4¤,2)+"."+RIGHT¤(A4¤,2)
460 A¤=A¤+A4¤:RETURN
470 REM RUECKGABE ALS A¤
480 REM ++++++++++++++++++++++++++++++++++++++++
490 REM *** UEBERTRAG MULTI ***
500 REM UEBERGABE UND RUECKGABE ALS A1/A2
510 A3=(A1-INT(A1))*100:A1=INT(A1)
520 A2=A2+(A3*100):A3=INT(A2/10000)*10000
530 A2=A2-A3:A1=A1+A3/10000
540 A8=A2:A9=INT(A8):A0=A8-A9
550 IF A0>=.5 THEN A4=A9+1:ELSE A4=A9
560 A2=A4:RETURN
570 REM ++++++++++++++++++++++++++++++++++++++++
580 REM *** MULTIPLIKATION ***
590 REM UEBERGABE FAKTOR 1  ALS A¤
600 REM UEBERGABE FAKTOR 2  ALS A6
610 GOSUB 100
620 A1=A1*A6:A2=A2*A6
630 A3=(A1-INT(A1))*100:A1=INT(A1)
640 A2=A2+A3*100
650 A3=INT(A2/10000)*10000:A2=A2-A3
660 A1=A1+A3/10000
670 A8=A2:A9=INT(A8):A0=A8-A9
680 IF A0>=.5 THEN A4=A9+1:ELSE A4=A9
690 A2=A4:GOSUB 370:RETURN
700 REM RUECKGABE ALS A¤
710 REM ++++++++++++++++++++++++++++++++++++++++
720 REM *** SUBTRAKTION ***
730 REM UEBERGABE ALS A1¤ UND A2¤
740 REM A2¤ WIRD VON A1¤ SUBTRAHIERT
750 REM A1¤ MUSS GROESSER
760 REM ALS A2¤ SEIN
780 A¤=A1¤:GOSUB 100
790 AC=A1:AD=A2:A¤=A2¤:GOSUB 100
800 AE=A1:AF=A2:AG=AC-AE
810 AH=AD-AF
820 IF AH<0 THEN AG=AG-1:AD=AD+10000:GOTO 810
830 A1=AG:A2=AH:GOSUB 370
840 REM RUECKGABE ALS A¤
850 RETURN
860 REM ++++++++++++++++++++++++++++++++++++++++
900 REM *** DEMONSTRATIONSBEISPIEL ***
1000 CLS:DIM Z(1,10):PRINT
1010 PRINT ">>> 3 ZAHLEN EINGEBEN <<<":PRINT
1020 INPUT" 1. ZAHL :";Z1¤
1030 INPUT" 2. ZAHL :";Z2¤
1040 INPUT" 3. ZAHL :";Z3¤
1050 REM *** ADDITION DER 3 ZAHLEN ***
1060 REM ZERLEGUNG UND SPEICHERUNG
1070 A¤=Z1¤:GOSUB 100
1080 Z(0,1)=A1:Z(1,1)=A2
1090 A¤=Z2¤:GOSUB 100
1100 Z(0,2)=A1:Z(1,2)=A2
1110 A¤=Z3¤:GOSUB 100
1120 Z(0,3)=A1:Z(1,3)=A2
1130 REM ADDITION UND UEBERTRAG
1140 A1=Z(0,1)+Z(0,2)+Z(0,3)
1150 A2=Z(1,1)+Z(1,2)+Z(1,3)
1160 GOSUB 330
1170 Z(0,4)=A1:Z(1,4)=A2
1180 REM SUMME ZUSAMMENSETZEN UND ANZEIGEN
1190 A1=Z(0,4):A2=Z(1,4):GOSUB 370
1200 PRINT:S¤=A¤
1210 PRINT"DIE SUMME BETRAEGT      :";S¤
1220 REM SUBTRAKTION DER 3.ZAHL VON DER SUMME
1230 A1¤=S¤:REM MINUEND
1240 A2¤=Z3¤:REM SUBTRAHEND
1250 GOSUB 780:REM SUBTRAKTION
1260 D¤=A¤:PRINT
1270 PRINT"DIE DIFFERENZ BETRAEGT :";D¤
1280 REM *** MULTIPLIKATION ***
1290 REM NACH VARIANTE 2
1300 PRINT
1310 PRINT"DIFFERENZ *  1.5       :";
1320 REM ZERLEGEN DER DIFFERENZ
1330 A¤=D¤:GOSUB 100
1340 REM AUSFUEHRUNG DER MULTIPLIKATION
1350 A1=A1*1.5:A2=A2*1.5
1360 GOSUB 510:REM UEBERTRAG
1370 GOSUB 370:REM ZUSAMMENSETZEN UND ANZEIGE
1380 PRINT A¤:PRINT
1390 REM NACH VARIANTE 1
1400 INPUT"BITTE FAKTOR X EINGEBEN:";A6
1410 REM A¤ ENTHAELT NOCH DAS 1.PRODUKT
1420 GOSUB 510
1430 PRINT"1.PRODUKT * FAKTOR X   :";A¤
1440 END
```

führt. Die beiden Zahlenteile werden den Variablen A1 und A2 übergeben, und mittels GOSUB 510 wird über das UP **Übertrag multi** lediglich der Übertrag ausgeführt. Das Ergebnis steht wieder in A1 und A2.

5. Subtraktion: Übergabe an das UP **Subtraktion** (GOSUB 780) erfolgt folgendermaßen:

Minuend als A1○
Subtrahend als A2○
VAL(A1○) > VAL(A2○)

Die Differenz wird als A○ zurückgegeben.

Hinweise zu den Über- und Rückgabevariablen, sowie eine Liste aller verwendeten Variablen sind als Kommentarzeilen in die Stringarithmetik eingeführt.

Beispielprogramm

Das Beispielprogramm enthält in den Zeilen 10 bis 860 die eigentliche Stringarithmetik, die zur Einhaltung einer zumutbaren Abarbeitungsgeschwindigkeit immer am Programmanfang stehen sollte. Im BASIC-Programm wird dann die Eingabe von 3 Zahlen gefordert, die zunächst zerlegt und dann addiert werden. Jetzt erfolgt die Subtraktion der 3. Zahl von der Summe. Die Differenz wird dann im BASIC-Programm mit dem Faktor 1,5 nach Variante 2 multipliziert.

Nach Eingabe eines weiteren Faktors durch den Bediener wird dann eine Multiplikation nach Variante 1 ausgeführt.

An diesem Beispiel müßte die grundsätzliche Verfahrensweise bei der Handhabung der Stringarithmetik zu erkennen sein. Bei sorgfältiger Auswahl der Variablen und übersichtlicher Arbeitsweise in der Programmierung ist eine leichte Einarbeitung in das Problem möglich. Mit der vorgestellten Variante lassen sich schon viele Probleme auf dem Gebiet der Ökonomie lösen, sie soll aber nicht als etwas Endgültiges betrachtet werden. Vielleicht findet der eine oder andere Leser weitere Lösungsmöglichkeiten bzw. kann die beschriebene Variante weiterentwickeln, um zum Beispiel auch negative Zahlen verarbeiten zu können, die Division zu realisieren, die Multiplikation zweier zerlegter Zahlen zu ermöglichen oder die Stellengenauigkeit weiter zu erhöhen.

Hans Langenhan, Ohrdruf

Magnetbandkatalog für KC 85/2,3 mit Zählwerk

Im Heft 12 (1987) dieser Zeitschrift wurde ein Programm zur automatischen Dokumentation der Programme auf den Magnetbandkassetten vorgestellt.

In diesem Beitrag wird eine Erweiterung des Programms dahingehend vorgenommen, daß auch der jeweilige Stand der Bandzähluhr automatisch mit erfaßt wird. Damit ist ein späteres Auffinden gewünschter Programme mühelos möglich. Die Idee des Programmes besteht darin, den freien Kanal 0 der CTC im KC 85/2,3 als Zeitgeber im Interrupt laufen zu lassen. Die RAM-Zellen 12300 – 12302 enthalten dann die jeweils abgelaufene Zeit T. Aus diesen Werten wird mit Formeln der Differentialgeometrie in den Programmzeilen 250 und 260 der jeweilige Bandzählerstand Q errechnet. Die angegebenen Werte beziehen sich auf das Tonbandgerät Geracord. Die Genauigkeit beträgt 0,5%. Sollten andere Bandgeräte oder 90-min-Bänder benutzt werden, so sind die entsprechenden Werte für P und Q in der Zeile 250 den folgenden Formeln zu zu entnehmen:

$$P = Q_o/(V-1)$$

und

$$Q = T \times (V^2-1)/T_o.$$

Dabei ist $V = r_2/r_1 = 0.459$ das Verhältnis der Wickelradien, falls das Zählwerk vom Abwicklerteller getrieben wird. Im anderen Fall ist $V = r_1/r_2 = 2{,}18$. Q_o ist der Endstand des Zählwerkes bei Banddurchlauf und T_o ist ein Maß für die Laufzeit, also $T_o = 5E+4$ für 60-min-Kassetten bzw. $T_o = 7.5E+4$ für 90-min-Kassetten.

Natürlich ist wiederum die Ausgabe auf einen Drucker durch Ersetzen des PRINT-Befehls in Zeile 430 durch PRINT # 2. möglich.

Wird die Bandversion des BASIC-Interpreters genutzt, so sind die Zeilen 530 und 540 zu ersetzen durch:

530 DATA 0,0,0,205,3,240,10,0
540 DATA 0,0,201,0,122,18,0,0

Prof. Dr. H. Junek

```
100 REM**********************
110 REM* HEADER MIT ZAEHLW. * (C)JUNEK
120 REM**********************
130 :
135 CLEAR 256,10000
145 GOSUB 500: REM MC-PGM
150 I=12288: REM #3000
160 CALL*3030: REM INIT INTERRUPT
170 Z=12300:FOR I=0 TO 2:POKE Z+I,0:NEXT I:!ZEITZELLEN
180 CLS:PRINT AT(31,03);INK 4;"EXIT=<BRK>+TB-SIGNAL"
190 DEF FNP(X)=VPEEK(X)+256*VPEEK(X+1)
200 PRINT " ZWS N A M E      TYP BEGINN LGE  AUTO"
210 CALL*3000:!KOPFBLOCK LESEN
220 IF PEEK(498)<>1 THEN210
230 T=PEEK(Z)+256*PEEK(Z+1)
250 P=389*(-1.647):Q=-15.5E-6*T:!389=ZW.-ENDE
260 Q=P*(SQR(1+Q)-1):Q=INT(Q+.5)
270 LET Z$=STR$(Q)+" "
280 IF LEN(Z$)<5 THEN Z$=" "+Z$:GOTO280
290 REM Z$=ZAEHLWERKSTAND
300 B=14080:REM B=BUFFER-#8000,KOPFBLOCK AUSWERTEN
310 I=0
320 TYP$=" COD"
330 IF VPEEK(B)=211 THEN I=I+3:TYP$=" PGM"
340 IF VPEEK(B)=212 THEN I=I+3:TYP$=" DAT"
350 !NAME
360 X=VPEEK(B+I):IF X=0 THEN X=32
370 Z$=Z$+CHR$(X):I=I+1:IF I<11 THEN360
380 IF TYP$<>" COD" THEN Z$=Z$+"    "
390 Z$=Z$+TYP$
400 IF TYP$<>" COD" THEN 430
410 A=FNP(B+17):E=FNP(B+19):S=FNP(B+21)
420 Z$=Z$+STR$(A)+STR$(E-A)+STR$(S)
430 PRINT Z$
460 GOTO210
490 REM MC-PGM
500 RESTORE:LET A=12288:REM #3000
510 READ X:IF X<0 THEN RETURN
520 POKE A,X:A=A+1:GOTO510
530 DATA 205,24,240,205,3,240,10,205
540 DATA 27,240,201,0,122,18,0,0
550 DATA 251,229,245,33,12,48,52,32
560 DATA 6,35,52,32,2,35,52,241
570 DATA 225,237,77,0,0,0,0,0
580 DATA 127,127,90,87,1,0,0,0
590 DATA 33,16,48,34,232,1,62,167
600 DATA 211,140,62,0,211,140,201,-1
610 RETURN
```



Zeitschriftenschau

Seit 1984 erscheint in der UdSSR die Zeitschrift „Mikroprozessornyje sredstwa i sistemy", herausgegeben vom Staatlichen Komitee der UdSSR für Wissenschaft und Technik. Wie G. Gromow, wissenschaftlicher Mitarbeiter des Rechenzentrums der AdW der UdSSR und verantwortlicher Sekretär der Zeitschrift in „Wissenschaft in der UdSSR", Heft 2/1987, schreibt, stehen im Blickpunkt der „MP" immer Probleme der Computer-Grundausbildung: Informatik in der Schule, Heimcomputer, Programmierunterricht an Hochschulen. Es finden sich Beiträge zu neuen Mikroprozessoren, Kleincomputern, Eigenbau-Heimcomputern mit Schaltbildern und Diagrammen sowie zur entsprechenden Software. Der Chefredakteur, A. Jerschow, erläuterte in der ersten Ausgabe der „MP", daß die Zeitschrift für jeden nutzbringend gemacht werden soll, der mit der Entwicklung und Anwendung der Mikroprozessortechnik zu tun hat. Beiträge aus unserer Schwesterzeitschrift dürften deshalb auch für den Leserkreis der MP (DDR) von Interesse sein, so daß wir unsere in loser Folge erscheinende Rubrik „Zeitschriftenschau" mit Rezensionen der MP (UdSSR) eröffnen.

L. I. Podolski, A. P. Ljaskowski: Programmiersystem Quasic-2 für Mikrorechner.
Mikroprozessornyje sredstwa i sistemy 4 (1987) 2
Quasic-2 ist ein Programmiersystem für Mikrorechner vom Typ Elektronika 60, DWK, SM 1300 u. ä., das Edition, Compilation, Test und Programmausführung unterstützt. Es besitzt folgende Leistungsmerkmale:
– Benutzung einer höheren Programmiersprache, die aber auch über maschinenorientierte Möglichkeiten verfügt. Die Sprache ist mit PLZ vergleichbar und an der Architektur der Elektronika 60 orientiert.
– Hohe Ausführungsgeschwindigkeit der Programme; wird durch Compilation erreicht.
– Einfaches Nutzerinterface, das sich durch Homogenität und die Möglichkeit des interaktiven Arbeitens auszeichnet. Unter Homogenität wird das Vorhandensein einer gemeinsamen Nutzeroberfläche für alle Systemkomponenten und eines einheitlichen Stils der Wechselwirkung Nutzer – System in allen Phasen der Programmerarbeitung verstanden.
Das interaktive Arbeiten ist durch Einfachheit und Komfort der Formulierungen der Systemanforderung und deren schnelles Abarbeiten gekennzeichnet.
– Geringe Anforderungen an die Rechnerkonfiguration
Die wesentlichen Vorteile (Interaktivität, schnelle Compilation, geringe Anforderungen an die Peripherie) werden dadurch erreicht, daß das System im Arbeitsspeicher resident ist. Daraus folgen natürlich Schranken für die Programmabmessungen. Bei 56 KByte sind etwa 10^3 Programmzeilen möglich.
Alle umgebungsabhängigen Bedingungen sind im Systemmodul und in den Driver konzentriert. Für das autonome Arbeiten unter dem Betriebssystem RAFOS und OSRW wurden Systemmodule und Driver geschaffen. (Typische Größen: Sytemmodul – 150 Zeilen Assembler, Driver – 50 bis 200 Zeilen)
Für grafische Terminals existiert ein Programmpaket. Das System Quasic-2 kann nur als Minimalkonfiguration eines modernen Programmiersystems angesehen werden. Diese sind durch das Vorhandensein von Werkzeugen für Entwurfsspezifikation, Validation und Wartung bzw. Programmgeneratoren gekennzeichnet.

G. R. Gromow: Die Spielkomponente des Personalcomputers: Stimulator der Kreativität, pädagogische Methode, Genre der Filmkunst.
Mikroprozessornyje sredstwa i sistemy 4 (1987) 3
Ausgehend von einer historischen Betrachtung wird die Konvergenz von Spiel, Lernprozeß und professionellem Training, die durch die massenhafte Anwendung von Personalcomputern einen Höhepunkt erreicht hat, konstatiert. Es wird festgestellt, daß die Spielkomponente den schöpferischen Prozeß entkrampft und stimuliert.
Der Autor, vor allem durch seine Arbeit „Die nationalen Informationsressourcen – Probleme der industriellen Nutzung" bekannt, unterscheidet drei Arten von Computerspielen: Positionsspiele, Dynamische Spiele und Spiele mit Schauwert.
Der Autor sieht im Computerspiel den Hauptweg zur Überwindung psychologischer Barrieren. Der Einfluß der Spielkomponente auf schöpferische Prozesse in Arbeit und Lehre ist mehr oder weniger klar. Ungeklärt ist der Einfluß auf die Kunst. Er macht sich mindestens in der Schaffung eines neuen Filmgenres, des Dialogfilms, bemerkbar. Dabei ist der Zuschauer nicht mehr zur passiven Aufnahme verurteilt, sondern kann durch Auswahl von Handlungslinien aktiv Einfluß auf die Fabel nehmen, womit er sich selbst in das Geschehen integriert.

A. L. Pashitkow: Die logische Struktur des Computerspiels.
Mikroprozessornyje sredstwa i sistemy 4 (1987) 3
Unter bestimmten Randbedingungen lassen sich drei funktionelle Komponenten des Computerspiels erkennen, die programmtechnisch realisiert werden müssen:
– Spielmilieu, d. h. die durch das Spiel imitierte „Welt"
– Wechselwirkung mit dem Spieler
– Wertung der Spielsituation.
Das Spielprogramm kann in zwei Komponenten unterteilt werden. Die erste realisiert die logische Struktur des Spieles, während die zweite den Spielprozeß auf dem Display abbildet. Bei der logischen Struktur unterscheidet der Autor drei Niveaus: operatives, taktisches und stategisches Niveau und gibt in PASCAL-ähnlicher Notion ein Beispiel für beliebige Spiele des betrachteten Typs.
Eine wesentliche Schlußfolgerung des Beitrages besteht darin, daß die besten bekannten Computerspiele sich dadurch auszeichnen, daß die Anforderungen auf alle drei Niveaus der logischen Struktur gleichmäßig verteilt sind.

Bockov, S. O., Smeljanski, R. L.: Ein symbolischer Debugger für C
Mikroprozessornyje sredstwa i sistemy 4 (1987) 5
Durch seine interaktive Arbeitsweise und die Tatsache, daß der Programmierer beim Testen sich auf C-Niveau bewegen kann, weist der beschriebene Debugger wesentliche Merkmale moderner Debugger auf.
Dabei benötigt der Debugger relativ wenig Speicherplatz (26 K). Er ist selbst in C geschrieben und auf der Elektronika 60 implementiert. Es erfolgte auch eine Implementierung auf den hierzulande wenig bekannten Personalcomputer „Corvet". Die grundlegenden Leistungsmerkmale des Debuggers sind:
– Möglichkeit der Programmausführung von einem beliebigen Programmpunkt aus
– Möglichkeit des Schrittbetriebes
– Möglichkeit der Trassierung markierter Operatoren
– Setzen von statischen und dynamischen Haltepunkten
– inverse Programmausführung
– Zugriff auf C-Datentypen und -strukturen
– Prüfung auf Konsistenz der formalen und aktuellen Parameter
– Einbetten von bedingten Debuggerkommandos in das C-Programm
– Editieren auf C-Niveau.
Die Grundidee bei der Realisierung des Debuggers besteht in der Übersetzung des C-Programmes in einen Zwischenkode relativ hohen Niveaus, der interpretiert wird. In diesem Zwischenkode erfolgt eine Trennung von Steueroperationen und Datenoperationen. Damit gelingt es, nach Programmänderungen den Aufwand zur Neuübersetzung äußerst gering zu halten. Der Steuerfluß ist im Zwischenkode durch Lapanor-Schemata dargestellt, die Datenoperationen in umgekehrt polnischer Notation.

Lemko, L. M., Gladkow, W. W., Jermakow, S. W., Shukow, W. N.: Der Mikrorechner Elektronika MK 85
Mikroprozessornyje sredstwa i sistemy 4 (1987) 4
Der Mikrorechner Elektronika MK 85 zeichnet sich durch seine Größe aus, die der eines Taschenrechners gleich ist. Die Programmiersprache ist BASIC. Die Programm- und Dateneingabe erfolgt über eine Polyfunktionaltastatur mit 54 Tasten. Pro BASIC-Zeile sind 63 Zeichen zugelassen. Davon werden jeweils als laufender Ausschnitt 12 Zeichen durch die Flüssigkristallanzeige dargestellt. Außerdem werden verschiedene den Programmablauf charakterisierende Signale dargestellt. Im Mittel kann die zulässige Programmlänge 150 Zeilen betragen.
Der Rechner ist auf Basis des 16-Bit-Prozessors T 243-2 aufgebaut. Der adressierbare Speicherbereich beträgt 64 KByte, die maximale Taktfrequenz 2 MHz. In der Rechnerversion werden 18 KByte benutzt, eine Speichererweiterung ist möglich. Es ist Batterie- und Netzspeisung möglich.
Der Einzelhandelspreis beträgt 135 Rubel. Damit steht dem sowjetischen Amateurprogrammierer ein BASIC-Rechner für erste Programmierversuche zur Verfügung, der vergleichbar mit westlichen Produkten ist. Jedoch der beschränkte Speicherbereich, die stark begrenzten Möglichkeiten des Festkristalldisplays und vor allem das Fehlen von externen Speichermedien werden den Nutzer bald zum Homecomputer BK-0010 greifen lassen.

Dr. Stiefel

Esperanto-Computerzeitschrift

Seit 1987 wird die Zeitschrift FOKUSO („Brennpunkt"; früher „Internacia Komputado" = internationale Computertechnik, -anwendung und -programmierung) vom ungarischen Esperanto-Verband in neuer Form und mit erweitertem populärwissenschaftlichem Inhalt herausgegeben.
Natürlich liegt der Schwerpunkt nach wie vor bei allen Themen, die mit Computern, insbesondere mit Heim- und Personalcomputern, zu tun haben wie:
– Programmierung, Expertensysteme, Sprachverarbeitung, Computergrafik
– Robotertechnik
– Sensortechnik
– Kybernetik
aber auch:
– Krebsforschung
– internationale Ökonomie
– Ornithologie.
All diese Artikel stammen, ganauso wie die zahlreichen humorvollen oder satirischen Zeichnungen, von Autoren verschiedener Länder. Selbst über die Computerproduktion der VR China wird berichtet. Besonderes Interesse findet bei den Lesern der jährliche Überblick über die Computerproduktion der RGW-Länder. Gleichzeitig wird damit die Tauglichkeit der nunmehr 100jährigen Plansprache selbst für solche hochaktuellen Themen eindrucksvoll unter Beweis gestellt.
Jedoch nicht nur für Leser, sondern auch für Autoren ist der internationale Charakter dieser Zeitschrift interessant, schließlich wird sie in mehr als 35 Ländern der Erde verkauft! Die Zeitschrift kann zum Jahrespreis von 24,– M (4 Hefte) bezogen werden über M. Behr, Koburger Str. 83, Markkleeberg, 7113.

Weber

Nutzerkatalog für KC 87

Mit der Ausgabe 1/88 erschien erstmals eine Sammlung bewährter Anwenderlösungen, die den Nutzern von Robotron-Kleincomputern zur Nachnutzung angeboten werden. Mit der Vorlage dieses Kataloges soll über bereits vorhandene effektive Hard- und Softwarelösungen informiert werden und somit Entwickungs- und Programmieraufwand beim Anwender eingespart werden. Die Wirksamkeit dieses Kataloges und seiner Fortschreibungen wird von der Kooperationsbereitschaft der Kleincomputernutzer abhängen.
Interessenten für diesen Katalog und Einsender nachnutzungsfähiger Lösungen wenden sich an:

VEB Robotron-Meßelelektronik „Otto Schön" Dresden, Abt. 1EKG, Postfach 211, Dresden, 8012

Systemsoftware – weiterhin ein Engpaß

Erst ein Jahr nach Produktionsaufnahme des Personalcomputers PS/2 durch die IBM Corp. können die Anwender die Basisversion des Betriebssystems BS/2 installieren. Die Standardversion soll erst 1989 ausgeliefert werden und die erweiterten Versionen, die den Communication-Manager und den Database-Manager enthalten, sollen zu einem noch späteren Zeitpunkt bereitstehen. Das Basisbetriebssystem soll erstmals den Protection-Level-Mechanismus nutzen, den die Intel-Prozessoren bieten.
Für das System PS/2-Modell 80 hat IBM das Unix-Betriebssystem AIX in einer ersten Version freigegeben. Die Komponenten, die Befehle und Hilfsprogramme für die Anwendungsentwicklung bereitstellen und Programme unter DOS 3.3-Steuerung lauffähig machen, sollen jedoch erst Ende 1988 zur Verfügung stehen. Mit dem Betriebssystem AIX ist IBM somit den Forderungen vieler Anwender nach Unix-Software gefolgt. Eine totale Hinwendung zu Unix ist jedoch nicht zu verzeichnen.
Die seit Jahren bestehende Diskrepanz zwischen der Bereitstellung der Hardware und der Systemsoftware wird an diesen Beispielen erneut offenkundig.
Diese Situation läßt den unabhängigen Softwarehäusern weiterhin ein breites Betätigungsfeld offen. Die Hardwarehersteller unternehmen ebenfalls große Anstrengungen, um auf dem Gebiet der Software und des Services Boden gutzumachen. Der Umsatzanteil für Software und Service soll von gegenwärtig etwa 35 % auf 50 %, bei IBM von 27 % auf 46 % erhöht werden.

Quelle: Computer-Ztg. – Leinfelder-Echterdingen 19 (1987) 25. – S. 1

Neuartige Solarzelle

An der amerikanischen Universität von Delaware wurde ein neuer Typ einer Solarzelle entwickelt, bei dem anstelle des bisher üblichen Siliziums ein neuartiges Keramik-Substrat benutzt wird. Damit soll diese Zelle um 90 % billiger sein als herkömmliche Zellen.
Auf dem Keramik-Substrat wird eine Schicht aus Siliziumkarbid erzeugt, deren hohes Reflexionsvermögen in den Zellen ausgenutzt wird. Darauf folgt eine Dünnschicht aus polykristallinem Silizium, die mit Epitaxiemitteln erzeugt wird. Hier findet die Wandlung von optischer in elektrische Energie statt.
Die ersten Versuchsmuster sollen einen Wirkungsgrad von rund 10 % haben. Mit weiteren Verbesserungen soll der Wert auf 17 % gesteigert werden können.

Quelle: Elektronik. – München 36 (1987) 15. – S. 7

Schalten in Femtosekunden

Der japanischen Firma Sumitomo Special Metals Corp. soll die Herstellung eines Schaltelements, das aus einer Blei-Lanthan-Zirkon-Titan-Kombination besteht und mit 100 Femtosekunden (1 Femtosekunde = 1 Billiardstel Sekunde) die kürzeste optische Shuttergeschwindigkeit hat, die jemals erreicht worden ist, gelungen sein. Der Werkstoff wurde warm isostatisch gepreßt und anschließend in einem speziell dafür entwickelten Prozeß gesintert. Die Sinterdauer betrug weniger als zehn Stunden. Der Transmissionsgrad war annähernd 100 bei einer Wellenlänge von rund 400 Nanometern.
Der Schaltkreis ist sehr klein und hat eine minimale Einfügungsdämpfung. Er soll sich für Kameraverschlüsse, Shutter-Arrays von optischen Druckern, dreidimensionale Bildgebersysteme und Anzeigegeräte eignen.

Wi.

aus Blick durch die Wirtschaft 14. 6. 87

GaAs-ICs billiger herstellen

Ein neues Verfahren zur Produktion von GaAs-ICs, mit dem die Stückkosten gesenkt werden könnten, wurde von der kalifornischen Firma Vitesse Semiconductor Corp. entwickelt. Dabei verzichtete Vitesse auf die sonst bei GaAs-ICs meist eingesetzte Goldmetallisation und auf die Luftisolierung im Chip, bei der komplizierte Brücken in den ICs aufgebaut werden müssen.
Die Firma setzt in ihrem Verfahren Aluminium und Aluminiumoxyd ein. Es sind acht Maskenschritte erforderlich. Die Ausbeute soll auf 28 bis 32 % gesteigert werden können, bezogen auf 6 000 bis 8 000 aktive Elemente pro Chip. In einem nächsten Schritt soll die Zahl auf 10 000 oder 20 000 aktive Elemente pro Chip erhöht werden.

Quelle: Elektronik. – München 36 (1987) 22. – S. 7

Superschneller Speicherchip

Das kalifornische Unternehmen Alliance Semiconductor hat nach eigenen Angaben einen sehr schnellen Speicherchip mit einer Kapazität von 1 MBit entworfen. Dieser dynamische Schreib-/Lesespeicher soll in Japan von dem Unternehmen NMB Semiconductor im zweiten Quartal 1988 gefertigt werden. Damit sollen die Speicher anderen leistungsfähigeren Bauelementen angepaßt werden.
Die neuen 32-Bit-Mikroprozessoren sind für Taktfrequenzen von 16 bis 20 MHz ausgelegt. Sie konnten ihre volle Leistung mit den bisher verfügbaren dynamischen Speichern nicht ausschöpfen. Ein Durchbruch soll mit dem neuen Speicherchip mit 15 MHz erreicht werden. Die Entwicklung einer Familie von 1 MBit dynamischen Schreib-/Lesespeichern in 1,2-μm-CMOS-Technologie soll zwischen beiden Firmen vereinbart worden sein.

Blick durch die Wirtschaft v. 24. 12. 87

Neues PC-Betriebssystem MS OS/2



Werkfoto

COMPAQ Computer Corporation gab im Februar die Einführung der Standard-Version 1.0 des in enger Zusammenarbeit mit Microsoft entwickelten Betriebssystems Operating System/2 – kurz MS OS/2 – für die COMPAQ Personal Computer mit den Intel-Prozessoren 80286 und 80386 bekannt.
MS OS/2 ist ein neues Single-User-(Einplatz) Multitasking-Betriebssystem für 80286- und 80386-Personal Computer. Es stellt dem Anwender einen größeren Speicheradreßraum zur Verfügung als die bisher verfügbaren Betriebssyssteme (statt 640 KByte jetzt 16 MByte). Die von COMPAQ veröffentlichte Standard-Version 1.0 von MS OS/2 ist vollkompatibel zu Anwendungen für die Standard-Ausgabe 1.0 des Betriebssystems OS/2 der IBM. Anwendungsprogramme laufen auf COMPAQ-Personal-Computern mit 80286- und 80386-Prozessor unter MS OS/2 mit höherer Geschwindigkeit als auf vergleichbaren Systemen.
Neue Anwendungsprogramme werden das gleichzeitige Ablaufen mehrer Programme und Tasks (Aufgaben) unter MS OS/2 Version 1.0 möglich machen. MS OS/2 wird dadurch bei einem wachsenden Kreis von PC-Anwendern zu einem wichtigen Faktor der Leistungssteigerung werden. Darüber hinaus ermöglicht MS OS/2 das rasche Wechseln von Anwendung zu Anwendung, das Austauschen von Daten zwischen verschiedenen Programmen und das Arbeiten mit wesentlich umfangreicheren Programmen als bisher.

Varistoren in Kapselstiften

Die Firma Solid State Division der General Electric/RCA hat sogenannte „Pin-Varistoren“ entwickelt, die um Anschlußstifte der IC-Kapseln angeordnet sind. Das soll zu Grundflächen-Einsparungen von max. 15 % führen können. Der eigentliche Entwicklungsfortschritt liegt darin, daß die Varistoren in ihren Abmessungen so verkleinert worden sind, daß sie einerseits Transienten gleich wie die herkömmlichen Typen unterdrücken können und andererseits Platz an den Kapselstiften finden.
GE/RCA sieht vorerst Anwendungsmöglichkeiten nur dort, wo die Miniaturisierung von Platinen hohe Priorität hat (MIL-Sektor). Zudem werden die neuen „Pin-Varistoren“ ungefähr um den Faktor 10 teurer sein als die bisher üblichen Typen

Quelle: Elektronik. – München 36 (1987) 22. – S.7

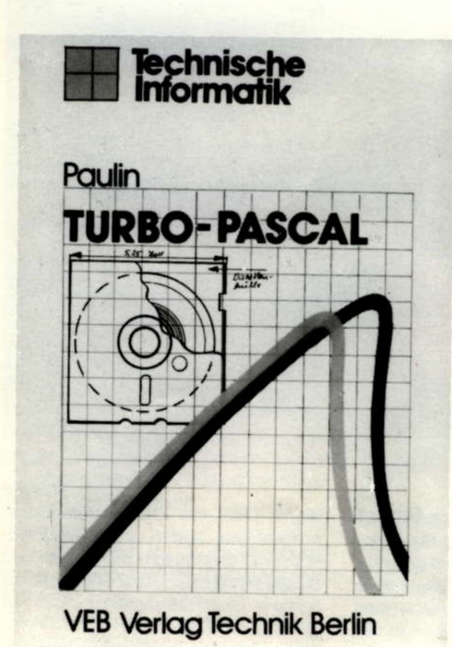
Si-Transistor: 50 ps Verzögerung

Neuartige bipolare Silizium-Transistoren, die versuchsweise in einem Ringoszillator eine Gate-Verzögerung von weniger als 50 ps (bei einer Flip-flop-Frequenz von 6,4 GHz) erreichen können, entwickelte die Firma Plessey/Großbritannien. Die Produktion der ersten ICs mit den neuen Transistor-Elementen soll 1988 aufgenommen werden.
Bei der Produktion der Transistoren benutzt Plessey einen eigenen „HE“-Prozeß, bei dem die sogenannte „Trench“-Isolation gemeinsam mit zwei überlappenden Lagen aus Polysilizium eingesetzt wird. Die Kapazität Emitter/Basis kann damit kleiner als üblich gehalten werden, und die nutzbare Betriebsfrequenz läßt sich anheben. Die Cutoff-Frequenz bei der Serienproduktion soll bei 14 GHz liegen.

Quelle: Elektronik. – München 36 (1987) 23. – S. 9

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 6
S. 161–192

Neuerscheinungen und Nachauflagen



TURBO-PASCAL

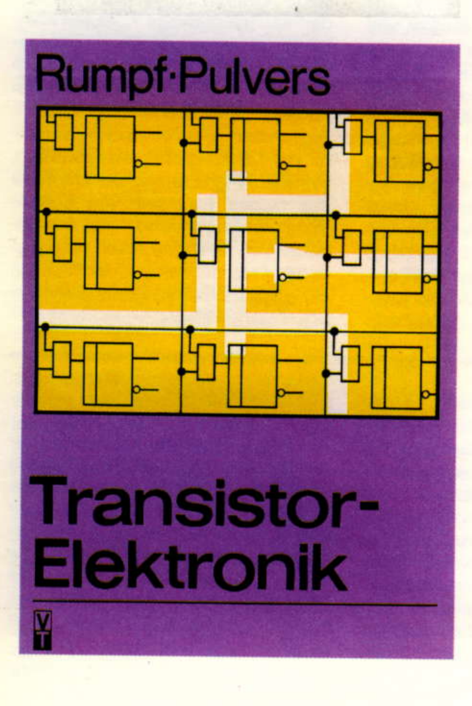
Von Dr. rer. nat. Gerhard Paulin. Reihe Technische Informatik. 1. Auflage. 208 Seiten, 20 Bilder, 3 Tafeln, Broschur, DDR 20,– M, Ausland 28,– DM. Bestellangaben: 553 921 0/Paulin, Turbo-Pascal
In dem Buch wird die Programmiersprache systematisch dargestellt, zahlreiche Beispiele illustrieren die Verwendung der Sprachelemente.
Darüber hinaus wird die Programmentwicklung im Dialog dargestellt: Quelltextbereitstellung, Compilation, Testläufe, Sichern von Quelltexten und Objektkodes. Die Anhänge enthalten Zusammenfassungen der Syntax, des Arbeitens mit dem Editor, Fehlernachrichten und der internen Kodierung (ASCII).

Optoelektronische Sensorsysteme

Von Prof. Dr. sc. techn. Detlef Schmidt und Dr.-Ing. Wolfgang Schwarz. Reihe Automatisierungstechnik, Band 231. 1. Auflage. 80 Seiten, 41 Bilder, 9 Tafeln, Broschur, DDR 4,80 M, Ausland 8,– DM. Bestellangaben: 553 930 9/Schmidt, Sensor RA 231
Am Beispiel des Einsatzes von CCD-Zeilenkameras wird gezeigt, wie durch zweckmäßige Informationsreduktion trotz großer Datenmengen ein Echtzeitbetrieb für Prüfprozesse möglich wird. Dabei wird auch den Fragen der zweckmäßigen Beleuchtung, der optimalen Rasterung und der geeigneten Merkmalauswahl Beachtung geschenkt.
Die zahlreichen, in einigen Fällen recht ausführlichen Anwendungsbeispiele sollen einerseits Anregungen geben für neuartige Lösungen in verschiedensten Produktions- und Überwachungsbereichen, andererseits auf besonders zu berücksichtigende Probleme hinweisen.

Grundlagen der elektrischen Meßtechnik

Von Prof. Dr. sc. techn. Werner Richter. 2., bearbeitete Auflage. 308 Seiten, 322 Bilder, 38 Tafeln, Leinen, DDR 19,– M, Ausland 32,– DM. Bestellangaben: 553 838 1/Richter, Meßtechnik
Neben einer gründlichen Darstellung von Kenngrößen und Kennfunktionen für Meßeinrichtungen sowie der gesamten Fehlerproblematik enthält das Buch vor allem die für die moderne elektrische/elektronsiche Meßtechnik typischen Funktionseinheiten, Meßgeräte und -systeme bis hin zu Sensoren und mikroprozessorgesteuerten Meßeinrichtungen. Viele Beispiele geben Hinweise für eine direkte Nutzung in der meßtechnischen Praxis.
Die Auflage ist aktualisiert worden.



Transistor-Elektronik

Anwendung von Halbleiterbauelementen und integrierten Schaltungen
Von Ing. Karl-Heinz Rumpf und Ing. Manfred Pulvers. 11., unveränderte Auflage. 296 Seiten, 334 Bilder, 43 Tafeln, Leinen DDR 24,– M, Ausland 35,– DM. Bestellangaben: 553 620 4/Rumpf, Transistor

Analoge Systeme

Grundlagen
Von Prof. Dr.-Ing. habil. Gerhard Wunsch und Dr. sc. techn. Helmut Schreiber. 2., durchgesehene Auflage. 238 Seiten, 220 Bilder, 7 Tafeln, Kunstleder, DDR 18,– M, Ausland 28,– DM. Bestellangaben: 553 896 3/Wunsch, Analoge Systeme
Von den gleichen Autoren bereits in 2., durchgesehener Auflage: Stochastische Systeme. Grundlagen. 176 Seiten, 111 Bilder, Leinen, DDR 13,– M, Ausland 25,– DM. Bestellangaben: 553 325 7/Wunsch, Stochast. Systeme

Auslieferung in diesen Tagen durch den Fachbuchhandel



VEB VERLAG TECHNIK BERLIN

ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK

MP

Heft 7 · 1988

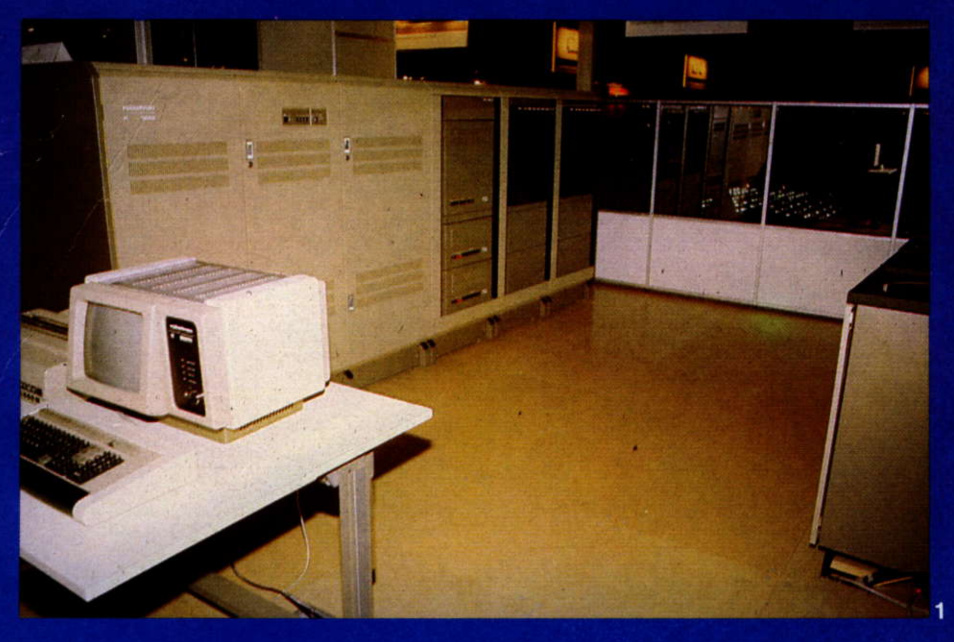
Mikroprozessortechnik

VEB Verlag Technik Berlin
ISSN 0233-2892

LFM '88

Steuereinheit für Festplattenspeicher

Erfahrungen mit einem lokalen Netz



1



2



Bilder von der Leipziger Frühjahrsmesse 1988



3



4



5



6



7



8

Lesen Sie dazu unseren Bericht in diesem Heft



9

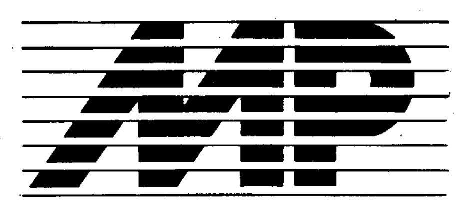


10



11

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 7 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2870371); Herbert Hemke, Redakteur (Tel. 2870203); Sekretariat Tel. 2870381

Gestaltung Christina Bauer

Titelfotos Hans-Joachim Pischeli (1), Hans Weiß (3)

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

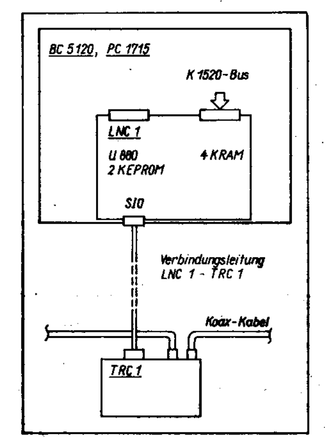
Redaktionsschluß: 16. Mai 1988

AN (EDV) 49837

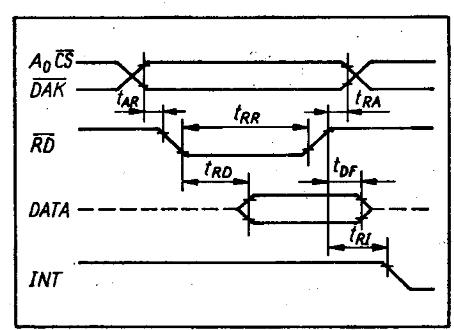
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

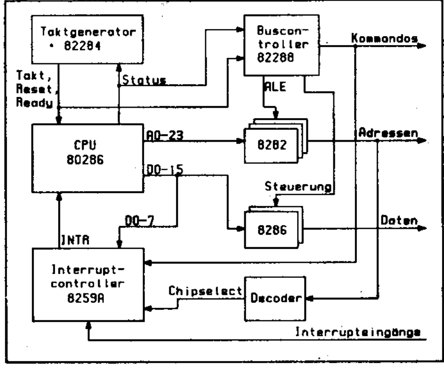
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propagandit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tísku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Há Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



Seite 199

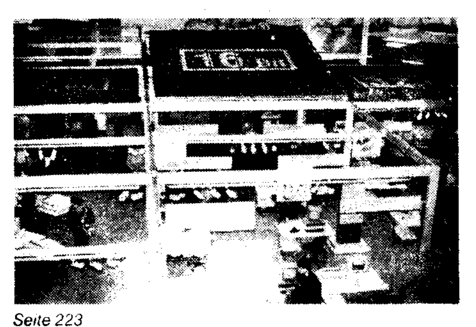


Seite 200



Seite 205

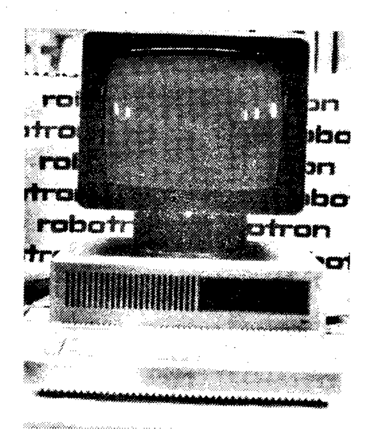
Leipziger Frühjahrsmesse 1988

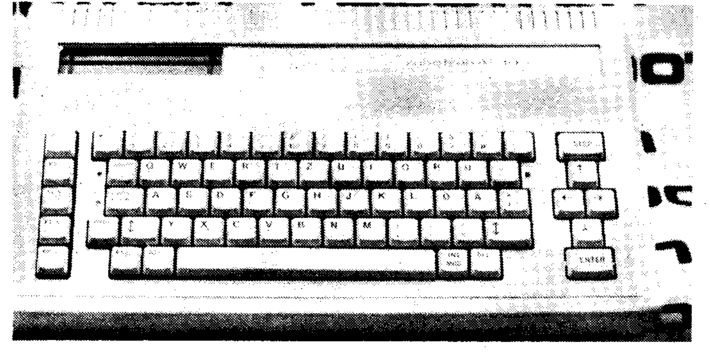


Seite 223

Inhalt

Neuer Robotron-Computer für Ausbildung und Freizeit





◀ 1 ▲ 2 Fotos: Hemke

Am 12. Mai dieses Jahres erlebten die Besucher eines Experimentalvortrages im deutschen Hygienemuseum eine Computerpremiere besonderer Art. Die URANIA und der VEB Robotron-Meßelektronik „Otto Schön" Dresden beschritten einen begrüßenswerten Weg bei der Vorstellung eines neuen Produktes. Den Zuschauern im vollbesetzten Steinsaal wurde der Bildungscomputer BIC A 5105 präsentiert. Über einen Schwarzweiß- und sechs Farbmonitore konnten die Zuschauer die Handlungen des Vorführenden am Computer verfolgen.
Der Computer (Bild 1) besteht aus drei Hauptbaugruppen: dem Computergrundgerät (Tastatur mit Computer; Bild 2), der Diskettenspeichereinheit und dem Monitor. Dieser Computer, dessen Entwicklung in diesem Jahr abgeschlossen wird, steht ab 1. September 1989 für die Ausbildung in Oberschulen, Berufs-, Spezial- und Ingenieurschulen zur Verfügung. Die Entwicklung des Computers erfolgt entsprechend den Forderungen unseres Bildungswesens, und die Produktionskapazität wird bedarfsdeckend für die genannten Bildungseinrichtungen vorbereitet. Nachdem der Bedarf des Bildungswesens gedeckt ist, soll eine abgerüstete Variante des Bildungscomputers, bestehend aus dem Computergrundgerät (Bild 2) und einem Beistellnetzteil in demselben Design, als Heimcomputer über den Einzelhandel angeboten werden.
Nach einem Einführungsvortrag vom Betriebsdirektor des VEB Robotron-Meßelektronik, Prof. Jugel, der sehr offen über alle technischen Details sowie die überwundenen und noch zu meisternden Schwierigkeiten sprach, wurde der Computer in allen technischen Parametern von Dr. Keller, Dr. Kleinmichel und Herrn Schreiber vorgestellt und vorgeführt. Danach beantworteten sie ausführlich Fragen aus dem Publikum. Hier ein paar technische Daten: Der Rechner arbeitet mit einer CPU UA 880 D und einem Videocontroller U82720 D. Er besitzt 64 KByte ROM-Speicher, 64 KByte Anwender-RAM sowie 64 K × 16 Bit Video-RAM. Der Anwenderspeicher wird auf maximal 1 MByte aufrüstbar sein. Als Monitore können 31-cm-Schwarzweißmonitore, wie der in Bild 2 gezeigte PC-1715-Monitor, 42-cm-Farbmonitore, wie z. B. der HCM 38-1 vom VEB Werk für Fernsehelektronik, sowie Farb- und Schwarzweißfernseher über RGB-, Video- und HF-Eingang angeschlossen werden. Mit einem Farbmonitor können z. B. 3 Bilder mit 640 × 200 Bildpunkten in 4 aus 16 Farben oder mit 320 × 200 Bildpunkten in 16 Farben dargestellt werden. Je nach Betriebssystem ist die Darstellung von 40 oder 80 Zeichen pro Bildschirmzeile in 16 Vorder- und 18 Hintergrundfarben möglich. Es können Töne auf 3 Kanälen mit 8 Oktaven erzeugt werden. Anschließbar sind über Centronics- bzw. V.24-Schnittstellen Drucker und Plotter. Der Rechner ist netzfähig (z. B. ROLANET 1) und erlaubt den Anschluß von 2 weiteren Diskettenlaufwerken (1.6), einem Kassettengerät, einer Maus sowie zwei Spielhebeln. Es existieren ein Schacht für KC-87-Module und ein Busanschluß. Als Betriebssysteme werden SCPX (vollkompatibel zum PC 1715) und das ROM-residente RBASIC (kompatibel mit BASI 1834) geliefert.

MP

Dritte neue Chipfabrik entsteht im Südosten Erfurts

1,5 Millionen hochintegrierte Speicherschaltkreise eines neuen Typs werden die 8500 Beschäftigten des VEB Mikroelektronik „Karl Marx" Erfurt in diesem Jahr zusätzlich für die Volkswirtschaft bereitstellen. Der 64-KBit-Speicher, der bei gleicher Fläche die vierfache Kapazität gegenüber seinem Vorgänger hat, ist unter anderem für Robotron-Computer und Automatisierungstechnik bestimmt. Insgesamt wird die Produktion unipolarer Schaltkreise in diesem Jahr im Stammbetrieb des Kombinates Mikroelektronik Erfurt auf 180 Prozent im Vergleich zu 1987 steigen. Über diese anspruchsvollen Vorhaben wurde Anfang April auf einer Parteiaktivtagung in der Thüringer Bezirksstadt informiert. Gemeinsam mit Gerhard Müller, Kandidat des Politbüros des ZK und 1. Sekretär der Bezirksleitung Erfurt der SED, berieten Vertreter aller am Aufbau des modernen Mikroelektronikzentrums Erfurt Südost beteiligten Betriebe, wie ein weiterer hoher Leistungszuwachs erreicht werden kann. Die Initiativen der Mikroelektroniker und Bauleute zielen vor allem auf ein hohes Tempo beim weiteren Aufbau und der Inbetriebnahme neuer Kapazitäten für die Halbleitervorfertigung sowie eines modernen Forschungs- und Entwicklungszentrums der Mikroelektronik. So ist vorgesehen, die dritte Produktionsstätte für hochintegrierte Bauelemente im kommenden Jahr vorfristig zum Probebetrieb zu übergeben.

ADN

Gute Bilanz

Bis Ende März wurden in der DDR 13757 Büro-, Personal- und Arbeitsplatzcomputer produziert; das sind 4488 Geräte mehr als im gleichen Zeitraum des Vorjahres. Bei Speichern für die Computertechnik beträgt der Zuwachs 18202 Stück und bei Druckern für Computer 8741 Stück. Die Produktion von technologischen Spezialausrüstungen für die Herstellung aktiver Halbleiterbauelemente war um 25 Prozent, von Lichtleiterkabeln um 28 Prozent und von monolithisch-integrierten Schaltkreisen um 38 Prozent höher als im gleichen Zeitraum des Vorjahres. Bei Robotern beträgt der Zuwachs 860 Stück. Im März waren in der Volkswirtschaft 47500 CAD/CAM-Arbeitsstationen und -systeme im Einsatz.

ADN

Robotron-Ausstellung in Budapest

Personal- und Arbeitsplatzcomputer sowie Schreib- und Drucktechnik zeigte eine Ausstellung des Robotron-Kombinats Anfang April in Budapest. Außenhandelsexperten der UVR sowie Vertreter von Anwendungsbetrieben konnten sich auch mit einem Bildverarbeitungssystem sowie einem frei programmierbaren Industrieroboter bekannt machen, der in der Produktion von Leiterplatten eingesetzt werden kann. Als neues Erzeugnis der ESER-Reihe wurde der Personalcomputer EC 1834 zum Einsatz in autonomen Datenbanksystemen oder zur Verkettung lokalen Systemen vorgestellt.
Der stellvertretende Generaldirektor des Robotron-Kombinates, Dr. Hans Speidel, erklärte anläßlich der Eröffnung gegenüber der Presse, daß das DDR-Kombinat im Geiste der jüngsten Gespräche der Ministerpräsidenten beider Länder seine Direktbeziehungen zu ungarischen Betrieben vertiefen will, wobei eine Produktions- und Vertriebskooperation angestrebt werde. Robotron biete auch eine Produktionskooperation auf dem Gebiet der Schreibtechnik an. Als Beispiele für den Einsatz von DDR-Rechentechnik in Ungarn führte er unter anderem das Videoton-Unternehmen an.

ADN

Personalcomputer in der ČSSR

Bis zum Jahre 1990 sollen in der ČSSR 300000 bis 350000 Personalcomputer eingesetzt werden, erklärte der Minister für Elektronische Industrie der ČSSR, Milan Kubat. Die elektronische Industrie will rund 150000 Mikrorechner herstellen, annähernd dieselbe Anzahl soll in Betrieben anderer Industriezweige produziert werden.
Der Produktion elektronischer Bauelemente für die tschechoslowakische elektrotechnische Industrie wird im achten Planjahrfünft der Vorrang eingeräumt. Bis zum Jahre 1990 soll sie auf das 2,5fache gesteigert werden. Um die Nachfrage nach den Bauelementen zu befriedigen, werden sie zum Teil aus anderen sozialistischen Ländern importiert.

ADN

Neue sowjetische Computerzeitschrift

Vor allem für PC-Anwender erscheint in der Sowjetunion die neue Zeitschrift „PC World USSR" mit Beiträgen zur Entwicklung, Applikation und Nutzung von PCs. Auch Produktvorstellungen, Industrieinformationen und Hinweise zu Technologietrends soll die Publikation enthalten.
PC World USSR erscheint vorerst vierteljährlich in einer Auflage von 50000 Exemplaren im Verlag Radio i Svjaz'. Wie dessen Direktor E. N. Salnikow erklärte, wird so bald wie möglich ein monatliches Erscheinen angestrebt. Auch rechne er damit, daß die Auflage der Zeitschrift, die an Zeitungskiosken und in Buchläden verkauft werden soll, nicht annähernd den Bedarf decken werde. In der Sowjetunion gebe es zur Zeit zwar nur etwa 100000 PCs, deren notwendige Anzahl schätze man jedoch auf etwa 30 Millionen.
Basis für die Herausgabe der in russischer Sprache gedruckten Zeitschrift ist eine gemeinsame Publikationsvereinbarung zwischen Radio i Svjaz' und der amerikanischen IDG Communications Inc., die im April in Framingham, USA unterzeichnet wurde.

Ähnliche Vereinbarungen gibt es bereits zur Herausgabe von Computerzeitschriften in der Volksrepublik China und in der Ungarischen Volksrepublik.
Neben der Beteiligung mit Arbeitskapital an dem Vorhaben wird die IDG Communications Inc. zur Unterstützung der redaktionellen Arbeit ein Desktop Publishing System der Dell Computer Corp. zur Verfügung stellen.

aus Computerworld-Számítástechnika 11/1988

Winchesterlaufwerk aus Brno

An der Entwicklung eines Winchesterlaufwerkes wird bei k.p. Zbrojovka Brno gearbeitet. Ein Entwicklungsmuster wurde zum „Tag des Personalcomputers", der vom ČSSR-Ministerium für Elektroindustrie am 18. November 1987 im Prager Kulturpalast durchgeführt wurde, vorgestellt. Dieses Festplattenlaufwerk mit der Bezeichnung C 7150 soll für den XT-kompatiblen PP 06 ab 1990 produziert werden. Es besitzt eine Kapazität von 27,5 MByte (unformatiert) und beinhaltet 3 Festplatten mit insgesamt 6 Magnetköpfen. Die Übertragungsgeschwindigkeit beträgt 5 MBaud und die Zugriffszeit 112 ms. Es wiegt 2,8 kg bei einer Größe von 82,5 × 146 × 209 mm^3 und einer Leistungsaufnahme von 18 VA.

aus ELEKTRONIKA 3/88

Neues von Videoton

Im April veranstaltete in Berlin die Ungarische Wirtschaftskammer in Zusammenarbeit mit der Kammer für Außenhandel, der Kammer der Technik und der Interwerbung die „Tage der ungarischen Wirtschaft und Technik", in deren Verlauf ein umfangreiches Vortragsprogramm geboten wurde. Das Spektrum reichte von Informationen zum ungarischen Wirtschaftssystem und zur Plankoordinierung über den Maschinenbau, die Rechentechnik bis zur Landwirtschaft und Biotechnologie.
Ein Beitrag gab dabei einen tieferen Einblick in das Produktionsprofil der Videoton-Werke, die, 1938 gegründet, 1971 mit der Serienproduktion der ersten Computer auf Lizenzbasis begannen und derzeit 20000 Mitarbeiter beschäftigen. Videoton ist damit heute der größte Betrieb Ungarns zur Computerproduktion. Erläutert wurde im Überblick die gegenwärtige und zum Teil künftige rechentechnische Palette, wobei besonders die Vorführung des VT 32 als leistungsstarke CAD-Station erwähnt werden muß. Geheimnis blieb noch, daß es einige interessante Produkte zur Budapester Messe im Mai geben würde – zum Beispiel den neuen 32-Bit-PC VT 180. (Über diesen Computer und weitere Erzeugnisse werden wir Sie in einem Messebericht informieren.)
Mehr für Fachleute gedacht, aber auch für potentielle Kunden, war das traditionelle Symposium für Videoton-Anwender in Klingemühle Anfang Mai. Neben Fachvorträgen zu den neuesten peripheren Geräten, Anwendungsbeispielen der VT-20- und anderen Mikrorechnern bildete einen Schwerpunkt die Vernetzung der Videoton-Computer.
Von Interesse waren hier die Koppelmöglichkeiten der R-11-, VT-32- und beliebigen ESER-Rechner, vor allem aber das Videoton ARCNET zur Vernetzung von PCs unter MS-DOS mit 2,5 MBit/s über max. 6 km. Im ARCNET kann jede Station User oder Server sein, im letzteren Fall ist PC-AT-Leistungsklasse Voraussetzung. Gegenwärtig wird das Token-Passing-Prinzip angewendet, jedoch soll bis Ende des Jahres Ethernet mit 10 MBit/s – am VT 32 und R 11 bereits realisiert – auch für die PCs in zwei Varianten zur Verfügung stehen. *MP*

Software contra Computervirus

Ein „Impfstoff" zur Bekämpfung des sogenannten Computervirus ist von der Sectra (Secure Transmission AB), einer schwedischen Computerfirma in Linköping, entwickelt worden. Dieser, wie es heißt, erste kommerziell erhältliche Schutz vor Verseuchung durch Computerviren soll es auch erleichtern, bereits angesteckte Systeme zu rekonstruieren.
Computervirus ist der populäre Name für Programme, die in die Software von ahnungslosen Anwendern eingeschmuggelt werden, um die dort gespeicherten Daten zu zerstören, zu ändern oder vollständig zu löschen. Der Virus hat die Fähigkeit, sich selbst zu vermehren. Wenn das normale Programm läuft, beginnt sich der Virus automatisch in alle anderen Programme zu kopieren, mit denen er in Berührung kommt. In den meisten Fällen ist es so gut wie ausgeschlossen, das gesamte System zu entseuchen. Die einzige Möglichkeit besteht darin, es mit Hilfe von Masterkopien der ursprünglichen Daten von neuem aufzubauen.
Das TCELL genannte Softwarepaket von Sectra soll imstande sein, EDV-Systeme zu überwachen und Fehler in der elektronischen Sicherung jeder Datei zu entdecken. Es wird normalerweise als ein Prozeß im System installiert, kann aber auch als Wache dienen, die den Operator warnt, wenn eine kontaminierte Datei entdeckt wird. TCELL überwacht nicht alle Dateien gleichzeitig, weil das die Leistungsfähigkeit des gesamten Systems beeinträchtigen würde. Stattdessen werden mehrere Dateien zufällig herausgegriffen und ständig verfolgt, andere hingegen nur von Zeit zu Zeit. In der Standardversion ist es als Software-Paket für VAX/VMS und UNIX erhältlich, kann aber auch für alle anderen Systeme maßgeschneidert werden. *ADN*

Sprechender Schulrechner für blinde Schüler entwickelt

Einen sprechenden Taschenrechner, der für den Mathematikunterricht in den Spezialschulen für blinde Kinder vorgesehen ist, haben Studenten der Sektion Informationstechnik an der TU Dresden entwickelt. Er entstand in enger Zusammenarbeit mit dem Blinden- und Sehschwachenverband der DDR und stellt ein Spitzenerzeugnis dar. Seine Grundlage ist der Schulrechner SR 1, der eine akustische Ausgabe erhielt. Dabei haben die Mitglieder einer Arbeitsguppe unter Leitung von Prof. Dr. Walter Tscheschner ein überaus kompliziertes Problem gelöst.
Der Rechner verfügt über einen Sprachsynthetisator. Seine Mikroelektronik vollzieht das nach, was der menschliche Kehlkopf vermag. Er erzeugt Schwingungen, die in ihrer Gesamtheit verständliche Laute ergeben. Kernstück dieser Neuentwicklung eines sprechenden Rechners ist ein Sprachsynthesemodul. Es wurde von zwei Studenten im Rahmen ihrer Diplomarbeit entwickelt. Die Steuerung des Synthesemoduls und seine Ankopplung an den SR 1 übernimmt ein Einchipmikrorechner. In übertragenem Sinne erhielt dadurch der „Kehlkopf" eine Verbindung zum „Hirn", die ihm sagt, was es dann sprachlich zu formulieren hat. Die Kompliziertheit der Aufgabe bestand vor allem darin, mittels technischer Hilfsmittel der menschlichen Stimme das abzulauschen, was der Rechner akustisch wiedergeben soll. Zu diesem Zweck mußten die Frequenzspektren der einzelnen menschlichen Laute analysiert und für ihre Wiedergabe technische Baugruppen geschaffen werden. Sie vermögen Frequenzgemische zu erzeugen, die über einen Rausch- oder Impulsgenerator zunächst Ursignale ergeben. Diese Signale gelangen in ein System von Filtern, die dann die jeweiligen Laute synthetisieren, sie zusammensetzen. Die gesamte Sprachsynthese wird durch eine Leiterplatte bewirkt. Einer weiteren, die ebenfalls ausschließlich aus elektronischen Bauelementen der DDR-Halbleiterindustrie besteht, obliegt die Steuerung des netzbetriebenen Rechners. Über einen in ihm angeordneten Miniaturlautsprecher hört der blinde Schüler, welche Aufgabe er dem Rechner eingegeben hat. Ziffernweise sagt ihm eine erstaunlich natürlich klingende männliche Stimme auch das Rechenergebnis an.
Rechner mit phonetischer Ausgabe sind international schon seit einigen Jahren bekannt. Sie gelten aber als wenig praktikable Lösungen für Blinde. Die nun vorliegende Entwicklung aus Dresden wird den Erfordernissen des Mathematikunterrichtes in den Spezialschulen für Blinde gerecht. Der Entwicklung liegen langjährige wissenschaftliche Erfahrungen zugrunde, die an der Sektion Informationstechnik der TU Dresden auf dem Gebiet der Sprachsynthese gewonnen worden sind. Vorbereitungen zur Aufnahme der Produktion des Rechners wurden inzwischen eingeleitet. *ADN*

Grafik-Computer aus Kuba

Der erste Grafik-Computer Kubas ist am zentralen Institut für Digitalforschung (ICID) Havanna entwickelt worden. Gemeinsam mit sowjetischen Spezialisten werden jetzt Programme für verschiedene Anwendungsbereiche erarbeitet. Die technischen Parameter des als CID 7060 bezeichneten Computers sollen internationalen Standards entsprechen.

Dialog

Zum Artikel „Programmieren der Funktionstasten des PC 1715 in BASIC" in MP 3/88, S. 94

Der Beitrag bedarf folgender Ergänzung:
Vom VEB Robotron-Büromaschinenwerk „Ernst Thälmann" Sömmerda wird das sehr leistungsfähige Software-Paket KEYINST [1], [2] vertrieben, das eine Umcodierung aller Tasten der Tastatur ermöglicht. Dabei entfällt der Umweg über BASIC, und das von KEYINST erzeugte Umcodierungsfile, *.KEY ist von der Betriebssystem-Ebene aufrufbar.

B. Sydoruk

Literatur

[1] KEYINST Tasteninstallation, Anwendungsbeschreibung und Bedienungsanleitung. VEB Büromaschinenwerk „Ernst Thälmann" Sömmerda, 1986
[2] Softwareangebot für PC 1715. VEB Büromaschinenwerk „Ernst Thälmann" Sömmerda, 1987

Zum Artikel „Ausdrucken von Bildschirminhalten beim PC 1715 unter SCP" in MP 3/88, S. 95

Herr Dagobert Mühlhaus stellte in seinem Beitrag eine Hardcopy-Routine für den PC 1715 vor, die aus jedem BASIC-Programm unter SCP aufgerufen werden kann. Dabei legte er folgende Formel zugrunde: „POSITION=63488+80*(Zeile-1)+Spalte".
Die dezimale Adresse 63488 enthält das erste Byte des Bildwiederholspeichers. Bei der Formel geht Herr Mühlhaus bei der Numerierung der Zeilen davon aus, daß die erste Zeile mit „1" beziffert wird, die erste Spalte aber mit „0". Da letzteres von ihm selbst nicht beachtet wurde, ist die Endadresse des Bildwiederholspeichers falsch gewählt und führt zum Ausdrucken einer nicht existierenden 25. Zeile. Die Endadresse muß richtig lauten: 65407.

D. Gralla

Steuereinheit für Festplattenspeicher

Bernd Däne, Dr. Wolfgang Fengler, Elke Thomä
Technische Hochschule Ilmenau, Sektion Technische und Biomedizinische Kybernetik

Der folgende Beitrag befaßt sich mit einem konkreten Mikroprozessor-Steuerungsproblem. Er ist aber methodisch verallgemeinerungsfähig als ein Beispiel für den Mikroprozessor-Geräteentwurf. Zugleich stellt der Festplattenspeicher ein zukunftsorientiertes, allgemein anwendbares Massenspeichermedium dar.

Einleitung

Gegenwärtig sind Magnetplattenspeicher ein wichtiges externes Speichermedium für Mikrorechnersysteme.
Festplattenspeicher (FPS; englisch: harddisk) erreichen bei Plattendurchmessern von 3¹/₂, 5¹/₄, 8 oder 14 Zoll Speicherkapazitäten von 10 bis 800 MByte, mittlere Positionierzeiten von 10 bis 100 ms und Transferraten von 4 bis 10 Mbit/s.
Die gegenwärtig weit verbreiteten Winchesterspeicher sind FPS, die durch folgende Besonderheiten gekennzeichnet sind:

– Die Schreib/Leseköpfe liegen beim Anlaufen und Anhalten auf der Plattenoberfläche auf.
– Bei Verwendung eines bestimmten Plattenmaterials fliegen die Köpfe bei Erreichen der Arbeitsgeschwindigkeit in einem Abstand von 0,35 bis 1 μm über der Plattenoberfläche.

Damit entfällt gegenüber früheren Systemen der Vorgang des Hebens und des Senkens der Köpfe sowie die dafür notwendige Mechanik. Der geringe Abstand zwischen Kopf und Platte ermöglicht eine hohe Aufzeichnungsdichte. Allerdings sind Magnetplatten mit einwandfreier Oberfläche und eine staubfreie Umgebung notwendig.

Dipl.-Ing. **Elke Thomä** studierte von 1972 bis 1976 an der Sektion Technische und Biomedizinische Kybernetik der TH Ilmenau und arbeitete anschließend als Assistent an dieser Einrichtung. Während dieser Zeit und der darauffolgenden Tätigkeit im VEB Robotron-Elektronik Zella-Mehlis arbeitete sie auf dem Gebiet der Steuerungstechnik und der Anwendung von Mikrorechnern. Seit 1987 ist sie als Ingenieur für Lehre und Forschung im Bereich Computertechnik der TH Ilmenau tätig.

Dipl.-Ing. **Bernd Däne** studierte von 1981 bis 1986 an der Sektion Technische und Biomedizinische Kybernetik der TH Ilmenau. Anschließend begann er eine Tätigkeit als wissenschaftlicher Assistent im Bereich Computertechnik dieser Sektion. Er bearbeitet Aufgabenstellungen zur Anwendung von Mikrorechnern.

Dr.-Ing. **Wolfgang Fengler** s. MP 1 (1987) 2, S. 38

Steuereinheit Anforderungen

Zum Betrieb von FPS ist eine Steuereinheit erforderlich. Diese hat die Aufgabe, den FPS mit dem Mikrorechner zu verbinden, die Daten zu übertragen und die Anpassung der verschiedenen Datenformate vorzunehmen. Als Schnittstelle zum Mikrorechner dient im allgemeinen der Systembus, über den Daten-, Adreß- und Steuersignale parallel übertragen werden. Gemäß der seriellen Aufzeichnung auf der Magnetplatte erfolgt die Datenübertragung an der Schnittstelle bitseriell. Daraus ergibt sich als Hauptaufgabe der Steuereinheit die Serien-Parallel- bzw. Parallel-Serienwandlung der Daten.

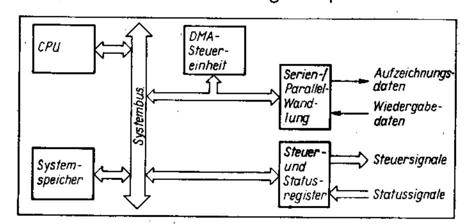


Bild 2 Steuereinheit mit DMA-Baustein

Schnittstellen zwischen Speichergerät und Steuereinheit

Bild 1 zeigt den prinzipiellen Aufbau einer Steuereinheit für FPS. Da der Ausstattungsgrad der FPS mit Ansteuerelektronik unterschiedlich ist, existieren auch unterschiedliche standardisierte Schnittstellen.
ST 506 ist das Standardinterface für Mini- und Mikrolaufwerke (3¹/₂ und 5¹/₄ Zoll). Aus den MFM-Daten (Modifizierte Frequenz-Modulation) müssen in der Steuereinheit beim Lesen durch eine Datenseparierung der serielle Datenstrom und der dazugehörige Takt zurückgewonnen werden.
Die Datenübertragungsgeschwindigkeit beträgt einheitlich 5 Mbit/s. Die Positionierung des Schreib-/Lesekopfes über die gewünschte Spur wird mit einer relativen Positioniersteuerung ausgeführt, d. h. bei jedem Impuls wird ein Schritt nach innen oder außen ausgeführt. Die Sektoren haben eine variable Länge. Der Sektoranfang wird mit einer Adreßmarke (flußwechselfreier Abschnitt) auf der Platte gekennzeichnet (Softsektorierung). **ESDI** (Enhanced Small Disk Interface) ist ein neuer Standard für kleine FPS. Die Daten werden nicht mehr im MFM-Format, sondern im NRZ-Code (No Return to Zero) geliefert.
Das **SMD**-Interface (Standard Module Drive) wird bei 8- und 14-Zoll-Geräten verwendet. Die Datenübertragungsgeschwindigkeit beträgt hier 4 bis 8 Mbit/s. Die Daten werden im NRZ-Code übertragen. Zur Positionierung des Schreib-/Lesekopfes wird dem FPS eine komplette Spuradresse übermittelt. Die Positionierung über die gewünschte Spur erfolgt dann selbsttätig. Der Standard läßt sowohl Hardsektorierung als auch Softsektorierung zu.
Das **SCSI**-Interface (Small Computer Systems Interface) wird eingesetzt, wenn die gesamte Ansteuerelektronik im FPS untergebracht werden kann. Damit wird der zusätzliche Hardwareaufwand im Mikrorechnersystem minimal.

Innere Struktur

Eine einfache Struktur für eine Ansteuereinheit ist in Bild 2 dargestellt. Mit Hilfe eines DMA-Bausteines wird der Datentransfer mit dem Systemspeicher abgewickelt. Da jedes gelesene Datenbyte sofort in den Systemspeicher gebracht werden muß (und umgekehrt), ergeben sich aus der Datenübertragungsrate des FPS hohe Zeitanforderungen an den Systembus. Die CPU muß den DMA-Baustein für jeden Such- und Übertragungsvorgang initialisieren und starten. Weiterhin ist die CPU durch die zahlreichen Steuervorgänge und Statusabfragen stark belastet. Deshalb ist diese bei Floppy-Disk-Steuerungen verbreitete Struktur zur Realisierung von Ansteuerschaltungen für FPS schlecht geeignet.
In der Regel wird in FPS-Steuereinheiten zur Entlastung des Systembusses ein Pufferspeicher verwendet, der mindestens einen Sektor zwischenspeichern kann. Zur Entlastung der System-CPU ist die Steuereinheit meist mit einer eigenen CPU ausgerüstet, die alle Initialisierungen, Steuervorgänge und Statusabfragen übernimmt. In Bild 3 ist eine solche FPS-Steuereinheit mit einem E/A-Prozessor dargestellt. Der E/A-Prozessor hat DMA-Fähigkeiten, mit deren Hilfe er den Datentransfer vom und zum Speicher abwickelt. Er übernimmt auch die Steuerung des FPS. Der übergeordnete Rechner übergibt lediglich „höhere" Kommandos wie

– Spur suchen
– Spur formatieren
– Datenblock schreiben
– Datenblock lesen.

Die Datenblöcke werden nicht der Reihe nach in aufeinanderfolgenden Sektoren abgelegt, sondern in verschachtelter Anordnung (vgl. Bild 5). Bei einer 5er-Verschachte-

Bild 1 Standardschnittstellen für Festplattenspeicher /3/

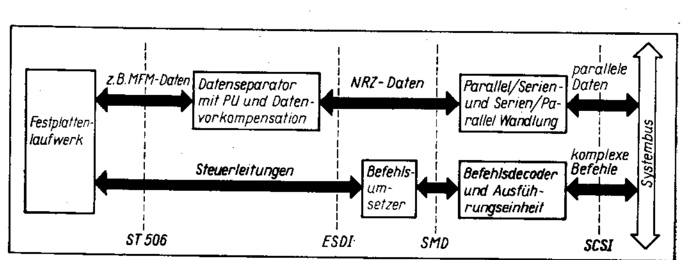
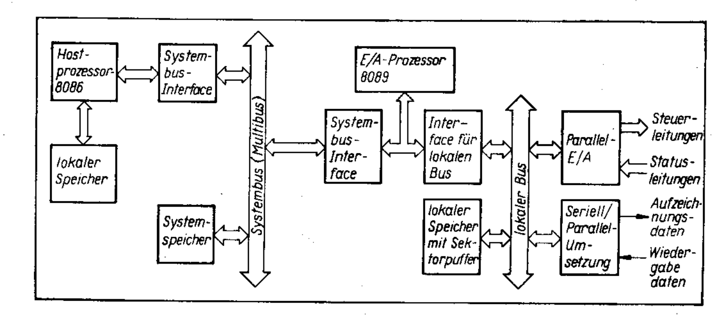


Bild 3 Steuereinheit mit E/A-Prozessor /2/



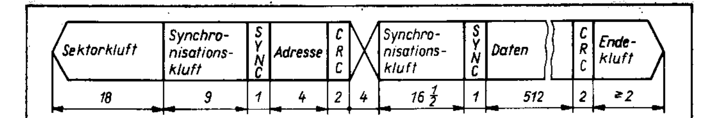


Bild 4 Sektorformat (Längenangaben in Byte)

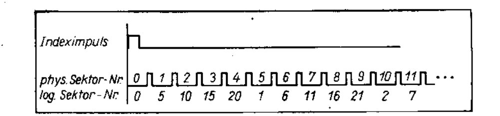


Bild 5 Verschachtelung der Sektoren

lung beispielsweise liegen zwischen zwei logisch aufeinanderfolgenden Sektoren vier andere Sektoren. Werden die Sektoren in ihrer logischen Reihenfolge angesprochen, so entsteht nach jeder Übertragung eine Pause, die für den Datentransfer zwischen Sektorpuffer und Systemspeicher genutzt werden kann. Deshalb müssen keine erhöhten Zeitanforderungen an den Systembus gestellt werden. Außerdem sind aufgrund der Pufferung der Daten Wiederholungen von Schreib- und Lesezugriffen sowie Fehlererkennungs- und Korrekturmaßnahmen ohne Beteiligung des übergeordneten Rechners möglich.

Steuereinheit für Festplattenspeicher mit SMD-Interface

Sektorformat

Die vorgestellte Steuereinheit unterstützt das Prinzip der Festsektorierung und realisiert ein Sektorformat mit 512 Datenbytes pro Sektor (Bild 4).

Das Adreßfeld ermöglicht eine Identifikation des Sektors. Ein vor dem Adreß- und dem Datenfeld eingefügtes Synchronisationsbyte („SYNC") ermöglicht der Steuereinheit beim Lesen einen definierten Start der Serien-Parallel-Wandlung. Mit Hilfe der CRC-Bytes wird die korrekte Übertragung des Adreß- bzw. Datenfeldes überprüft.

Die Sektoren sind entsprechend einer 5er-Verschachtelung numeriert (Bild 5).

Funktionsübersicht

Die Steuereinheit führt in Zusammenarbeit mit dem FPS die in Tafel 1 aufgeführten elementaren Funktionen aus. Es handelt sich dabei um Steuerfunktionen für die Bedienung des FPS sowie um elementare Schreib- und Lesevorgänge. Ein Teil davon ist nur für Inbetriebnahme- und Testzwecke von Bedeutung.

Zur Ausführung von komplexen Kommandos werden durch die Software Ablauffolgen aus diesen Funktionen gebildet.

Beschreibung der einzelnen Funktionsgruppen

Die beschriebene Steuereinheit realisiert eine Struktur mit Pufferspeicher und eigener CPU (Bild 6). Sie besteht aus vier Modulen im K-1520-Format.

Die **Mikrorechnereinheit** stellt eine Konfiguration auf Basis der Schaltkreisfamilie UB 880 (für Übertragungsraten bis 5 Mbit/s) bzw. UA 880 (für Übertragungsraten bis 8 Mbit/s) dar.

Die CPU organisiert den Datentransfer zwischen FPS und Zweitorspeicher gemäß den vom übergeordneten System im Zweitorspeicher übergebenen Befehlen. Der Einsatz von zwei DMA-Schaltkreisen macht sich wegen der hohen Geschwindigkeitsanforderungen erforderlich.

Der statische RAM (1 KByte) dient als Arbeitsspeicher und als Puffer für die Daten, die vom bzw. zum Plattenspeicher übertragen werden sollen.

Ein CTC-Schaltkreis U 857 unterstützt die Ablaufsteuerung in der Datenwandlereinheit.

Tafel 1 Elementare Funktionen der Steuereinheit

Steuerfunktionen
Initialisieren: Herstellen des Grundzustandes von Steuereinheit und Plattenspeicher
Speicher Ein/Aus: physisches Ein- und Ausschalten des Plattenspeichers
Auswählen Gerät: logische Freigabe eines angeschlossenen Plattenspeichers
Geräteregister Löschen: Löschen des Fehlerregisters im Plattenspeicher (Quittieren einer Fehlermeldung)
Nullstellen: Positionieren der Köpfe auf Zylinder Null
Positionieren: Positionieren auf den adressierten Zylinder, Einschalten des adressierten Kopfes
Offset-Positionieren: geringfügiges Verschieben des Kopfes aus der Spurmitte und/oder Verschiebung des Datenübernahmeimpulses
Aufzeichnungs- und Wiedergabefunktionen
Wiedergabe Daten: Lesen des Datenfeldes eines Sektors
Aufzeichnen Daten: Schreiben des Datenfeldes eines Sektors
Aufzeichnungskontrolle: Lesen des Datenfeldes eines soeben geschriebenen Sektors
Wiedergabe Adresse: Lesen eines Adreßfeldes der positionierten Spur
Wiedergabe Sektor: Lesen eines kompletten Sektors mit Adreß- und Datenfeld sowie Klüften
Aufzeichnen Adressen: Formatieren einer Spur

Der **Zweitorspeicher,** der wahlweise mit U 256 oder U 2164 (jeweils 8 Stück) bestückt werden kann, dient als Bindeglied zum Systembus des übergeordneten Rechnersystems.

Der Refresh für den dynamischen RAM wird wahlweise durch die CPU oder durch einen DMA der Steuereinheit durchgeführt /6/, /7/.

Der beidseitige Zugriff zum Zweitorspeicher wird durch eine Zugriffslogik gesteuert.

In der **Datenwandlereinheit** sind mehrere Funktionsgruppen realisiert.

Die *Zeitgeber- und Steuerlogik* erzeugt in Verbindung mit dem CTC-Schaltkreis der Mikrorechnereinheit eine Reihe von Signalen mit programmierbarem Zeitverlauf für die Steuerung der Übertragungsvorgänge.

Die *Taktaufbereitung* stellt alle für die Steuereinheit erforderlichen Takte bereit und realisiert deren Umschaltung.

Die *Ready-Steuerung* steuert die Busübernahme der DMA-Bausteine und paßt deren Übertragungsgeschwindigkeit an die Datenrate des FPS an.

Der *Parallel-Serien-Wandler* formt den byteparallelen Datenstrom des Mikrorechners in einen seriellen Datenstrom um und umgekehrt.

Mit dem *Digitalkomparator* wird der Sektoradreßvergleich realisiert. Bei der Suche nach einem bestimmten Sektor werden die einzelnen Bytes der gesuchten Sektoradresse nacheinander im Schreibpuffer bereitgestellt, während die gelesenen Adreßbytes nach der Serien-Parallel-Wandlung in den Vergleichspuffer gelangen. Bei Übereinstimmung aller Adreßbytes wird die Datenübertragung ausgelöst.

Der *SYNC-Vergleicher* sucht beim Lesen im seriellen Datenstrom das Synchronisationszeichen, das einen definierten Start der Serien-Parallel-Wandlung ermöglicht.

Die *CRC-Logik* enthält ein rückgekoppeltes Schieberegister, welches nach der SDLC-Vorschrift für jeden Datenblock zwei Kontrollbytes berechnet. Diese werden beim Schreiben automatisch an den Datenblock angefügt. Beim Lesen erfolgt eine Vergleichsberechnung, die über die Richtigkeit der gelesenen Daten Aufschluß gibt.

Die **Interfaceeinheit** realisiert den Anschluß des FPS an die Steuereinheit. Die Verbindung zum FPS erfolgt über zwei Interfacekabel. Aufgrund der hohen Übertragungsgeschwindigkeit und der notwendigen Störsicherheit sind alle Leitungen angepaßt und symmetrisch getrieben. Die rechnerseitige Ankopplung erfolgt über Treiberbausteine und über einen PIO-Schaltkreis.

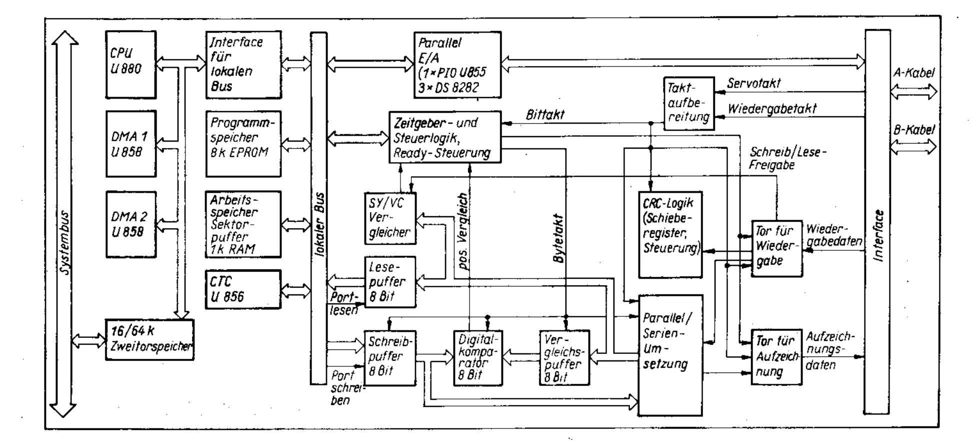
Softwarebeschreibung

• Schnittstelle zum übergeordneten Rechner

Die Software besteht aus verschiedenen Moduln in drei Hierarchieebenen (vgl. Bild 7).

Die obere Ebene, der „Kommandointerpreter", organisiert den Dialog mit dem übergeordneten Rechner (ÜR). Der Ablauf des Dialogs wird mit Hilfe von Synchronisationszellen geregelt.

Bild 6 Blockschaltbild für eine Steuereinheit mit eigener CPU auf U 880-Basis



im Kommandoregister übergibt der ÜR seine Kommandos an die Steuereinheit. Im Datenübergabebereich werden die Datenblöcke (je 512 Byte) übergeben.
Im Statusregister übergibt die Steuereinheit Statusmeldungen, die dem ÜR die Beendigung der Kommandoausführung und eventuell aufgetretene Fehler anzeigen.
An der Schnittstelle zum ÜR existieren virtuelle Blocknummern, die alle existierenden Sektoren fortlaufend numerieren, ohne auf die Spureinteilung Rücksicht zu nehmen.

• Aufgaben des Kommandointerpreters

Der Kommandointerpreter realisiert den Dialog mit dem ÜR und die Ausführung von dessen Kommandos auf der Grundlage der in Tafel 1 aufgeführten elementaren Funktionen. Die dabei verwirklichten Funktionsabläufe können relativ komplex sein, so daß eine spürbare Entlastung des ÜR eintritt.
Die Arbeit des Kommandointerpreters ist durch folgende Funktionsmerkmale gekennzeichnet:

- Zwischenspeicherung einer größeren Anzahl von Datenblöcken im Zweitorspeicher für den Fall einer nochmaligen Verwendung
- Führen eines Inhaltsverzeichnisses über diesen Bereich
- Verwirklichen einer sinnvollen Strategie für die Auswahl der zwischenzuspeichernden Datenblöcke
- komfortable Fehlerbehandlung: Eine vom Elementarniveau empfangene Fehlermeldung wird erst dann weitergegeben, wenn wiederholte Versuche zur Fehlerbeseitigung (mit entsprechenden Steuerkommandos) erfolglos blieben.
- Verwirklichung einer zeitoptimierten Strategie beim Schreiben von Datenblöcken, wobei während des Schreibens bereits weitere Kommandos übernommen werden können und beim Schreiben vorgemerkter Blöcke eine bezüglich der Positionierungszeiten optimale Reihenfolge gewählt wird.

Bei Bedarf ist auch ein direkter Zugriff zu den Funktionen des Elementarniveaus möglich. Dabei entfallen alle geschwindigkeitssteigernden Maßnahmen.
Es existieren auch Kommandos für Testzwecke, wie z. B. die Wiedergabe von Adreßfeldern und kompletten Sektoren, sowie verschiedene komplexe Kommandos, die sich mit dem Überprüfen, Formatieren und Füllen der Sektoren auf einer, mehreren oder allen Spuren befassen.

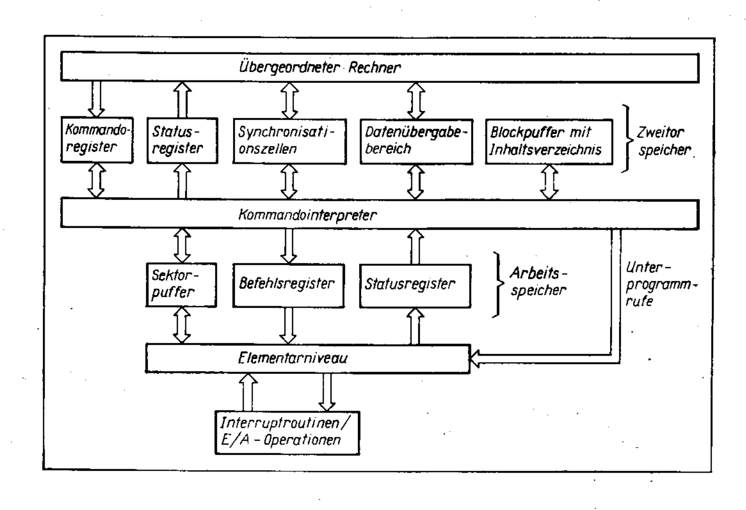


Bild 7
Softwarestruktur der Steuereinheit

• Aufgaben des Elementarniveaus

Im Programmsegment „Elementarniveau" wird die Ausführung der elementaren Laufwerksfunktionen organisiert. Die Aktivierung der einzelnen Kommandos erfolgt im Kommandointerpreter durch Aufruf eines entsprechenden Unterprogramms des Elementarniveaus.
Die Parameter und Daten befinden sich im Arbeitsspeicher. Nach Abschluß der Kommandoausführung werden eine Fertigmeldung und Informationen über eventuell aufgetretene Fehler an den Kommandointerpreter übergeben (vgl. Bild 7).
Die Aktivitäten des Elementarniveaus sind zum großen Teil als Interruptroutinen organisiert, wobei die Interrupts aus dem Zeitregime des FPS abgeleitet werden. Im Hintergrund dieser Interruptroutinen kann der Kommandointerpreter bereits weiterarbeiten.

Schlußbemerkungen

Die beschriebene Ansteuereinheit wurde auf 4 Leiterplatten im K-1520-Format realisiert. Zur Zeit wird sie am K-1520-Bus zusammen mit einem FPS vom Typ K 5501.03 getestet. Eine Anpassung an das Bussystem des MMS 16 (I-41-Bus /8/, /9/) und Einbindungen in vorhandene Betriebssysteme sind in Vorbereitung.

Literatur

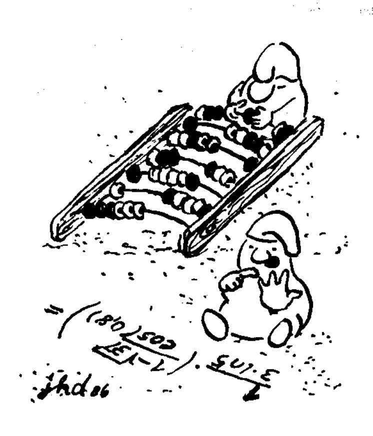
/1/ Thurn, K.: Steuerung von Magnetplattenlaufwerken. Elektronik (1983) 4, S. 49
/2/ Hofer, R.: Ein/Ausgabeprozessor als Controller für Winchester-Plattenlaufwerk. Elektronik (1983) 19, S. 85
/3/ Zeltwanger, H.: Chipsätze zur Ansteuerung von Winchesterlaufwerken. Elektronik (1985) 21, S. 81
/4/ Weiss, R.: Winchestertechnik: Massenspeicher mit Zukunft. Elektronik (1983) 19, S. 75
/5/ Lichtenauer, G.: Entwurf und Realisierung einer Festplattenansteuerung für ein Mikrorechnersystem. Diplomarbeit, TH Ilmenau, 1985
/6/ Thomä, E.; Fengler, W.; Däne, B. u. a.: Schaltungsanordnung zur Regenerierung dynamischer Halbleiterspeicher. WP G 06 F / 296 117 6
/7/ Däne, B.; Thomä, E.; Fengler, W. u. a.: Verfahren zur Regeneriersteuerung von dynamischen Zweitorspeichern. WP G 06 F / 301 729 2
/8/ I-41-Standard auf Grundlage des Normativmaterials der mehrseitigen Regierungskommission für die Zusammenarbeit auf dem Gebiet der Rechentechnik der sozialistischen Länder (Oktober 1984)
/9/ Mikrocomputer-Baugruppensystem AMS, Systemübersicht 1987, Firmenschrift Siemens

KONTAKT

Technische Hochschule Ilmenau,
Sektion Technische und Biomedizinische Kybernetik,
Bereich Computertechnik, PSF 327, Ilmenau, 6300;
Tel. 74 843 (Thomä)

Kleines Lexikon der Mikrorechentechnik

Wie sich künstlerische Freizeitbeschäftigung auf spezielle Art mit dem Beruf verbinden kann, zeigt die hiermit beginnende kleine Karikaturserie von Jens-Helge Dahmen. Das „Kleine Lexikon der Mikrorechentechnik" entstand in Auseinandersetzung mit seinem Beruf als wissenschaftlicher Mitarbeiter der Akademie der Wissenschaften der DDR, Institut für Informatik und Rechentechnik. Die Karikaturen von Jens-Helge Dahmen sind nur ein Teil seines bildnerischen Volksschaffens. So erweckte er schon 1986/87 mit Materialcollagen, die in verschiedenen Zeitschriften veröffentlicht wurden, die Aufmerksamkeit der Leser. In der „Gruppe Rot" des Studios „Otto Nagel" in Berlin widmet er sich der Malerei und Grafik. Besonderen Anklang bei den Mitarbeitern der Akademie der Wissenschaften der DDR findet eine jährlich im Rahmen der Kreis-MMM der Akademie der Wissenschaften der DDR stattfindende kollektive Ausstellung von Volkskunstschaffenden des Instituts für Informatik und Rechentechnik, deren Mitinitiator Jens-Helge Dahmen ist.



◄ Arithmetisch logische Einheit



Befehlszähler ▲

Erfahrungen mit einem lokalen Netz

Dr. Kurt Grubba,
Hans-Martin Adler
Akademie der Wissenschaften der DDR, Zentrum für wissenschaftlichen Gerätebau

Einleitung

Lokale Netze (LAN), die den Verbund rechentechnischer Einrichtungen innerhalb eines Verantwortungsbereiches zum Ziel haben, gewinnen immer mehr an Bedeutung /1/.
Vom Kombinat Robotron werden gegenwärtig die ROLANET-1-Baugruppen

- Local Network Controller (LNC 1) und
- Transceiver (TRC 1)

in die Produktion überführt. Diese Hardwarekomponenten sind zum Aufbau lokaler Netze mit Büro-, Personal- und Arbeitsplatzcomputern vorgesehen /2/. Mit Entwicklungsmustern sowie den Arbeitsversionen der zugehörigen Softwarekomponenten SCPNET und NETSERV wurden auf dem Gelände Berlin-Adlershof der AdW der DDR zwei experimentell-produktive Netze für 18 Stationen mit PC 1715 und BC 5120 im Institut für Informatik und Rechentechnik (IIR) und für 30 PC 1715 im Zentrum für wissenschaftlichen Gerätebau (ZWG) aufgebaut.

Gerätetechnische Ausführung

Mit ROLANET 1 steht ein System für die Konfiguration von lokalen Netzen mittlerer Leistungsfähigkeit zur Verfügung (Tafel 1). Im Vergleich zum internationalen Stand ist die Übertragungsrate im unteren Teil des Leistungsangebotes angesiedelt. Sie ist durch die gewählte Bauelementebasis des U 880-Systems bedingt, kann aber für die vorgesehenen Kommunikationsanforderungen (Dialog, Filetransfer) als ausreichend angesehen werden. Die Baugruppe LNC 1 (im K-1520-Format) wurde beim BC 5120 als zusätzliche Steckeinheit untergebracht, beim PC 1715 anstelle der optionalen Interfaceanschlußsteuerung eingesetzt, wobei die Verbindung zur internen Schnittstelle X1 über ein Adapterkabel erfolgte. Elektrische Veränderungen sind bei den BC/PC nicht erforderlich, so daß die Aufrüstung schnell und problemlos ausführbar ist.
Der Transceiver TRC 1 wird in das Koaxialkabel eingefügt und über eine mindestens 10polige Leitung mit dem LNC 1 verbunden (Bild 1). Benutzt wurden ein vorhandenes Koaxialkabel 50–7–2 und eine Fernmeldeplastleitung.

Einsatzvorbereitung

Bei der Projektierung der Netze wurde davon ausgegangen, vorhandene Kabelkanäle und -trassen für Fernmeldeleitungen mitzunutzen. Dadurch ergaben sich zwar größere Kabellängen (IIR: 1 km, ZWG: 0,7 km), aber aufwendige Schacht- und Maurerarbeiten entfielen weitgehend.
Um Flexibilität gegenüber Strukturveränderungen der Arbeitsplätze zu gewährleisten, wurden die Standorte der Transceiver so gewählt, daß ein flexibler Anschluß der möglichen Arbeitsplätze erreicht werden kann.
Parallel zum Aufbau des Netzes im ZWG wurde ein Organisationsprojekt erarbeitet. Es enthält u. a. die Aufgabengebiete (Tafel 2) und Festlegungen, welche Teilnehmer dabei als Sender, Sender und Empfänger oder nur als Empfänger fungieren. Für die Pflege zentraler Datenbestände wie Artikelstammdateien ist eine ständige Aktualisierung festgelegt.
Eine Analyse des derzeitigen Informationsflusses im ZWG zeigte, daß die durchschnittliche Ergiebigkeit der Informationsquellen für die innerbetriebliche Kommunikation zwischen 2 und 110 Normseiten (2000 Zeichen/Seite) je Tag und Struktureinheit liegt. Bezogen auf die 30 PC des ZWG wird der gesamte Informationsfluß auf 10 MByte je Tag geschätzt. Die max. Nettoübertragungsrate von ROLANET 1 liegt bei 500 MByte/Schicht, so daß selbst bei den zu erwartenden steigenden Kommunikationsanforderungen statistisch betrachtet zusätzliche Zeitverzögerungen durch Überlastung unwahrscheinlich sind.

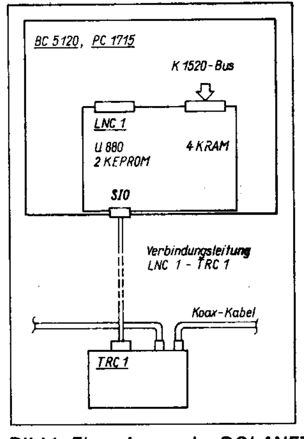


Bild 1 Einordnung der ROLANET 1-Baugruppen

Tafel 1 Einordnung von ROLANET 1 (halbfett) in international übliche Merkmale von PC-Netzen

Übertragungsmedien:	*verdrillte Cu-Leitung,* ***Koaxialkabel,*** *LWL-Kabel*
Topologie:	*Stern,* ***Bus,*** *Ring*
Übertragungsrate:	*0,01 ...* ***0,5*** *... 10 MBit/s*
Zugriffsverfahren:	*Polling,* ***CSMA/CD,*** *Token*
Übertragungsverfahren:	***Basisband,*** *Breitband*
Netzlänge:	*0,5 ...* ***1*** *... 3 km*
Teilnehmer:	*32 ...* ***100*** *... 255*

Tafel 2 Aufgabengebiete für das ZWG-Netz

1. *Bürokommunikation (Routinenachrichten)*
2. *Leitungsinformationssystem*
   - *Terminkontrolle*
   - *Planungskennziffern*
   - *Haushaltsabrechnung*
3. *Wissenschaftlich-technische Informationsfonds*
   - *Datenbasen zum wissenschaftlichen Gerätebau*
   - *Neuererwesen, Nachnutzungen*
   - *Standards*
   - *Literatur*
4. *Technische Vorbereitung der Fertigung*
   - *Datenbasen für Tevo*
   - *Materialwirtschaft*
   - *Fertigungsaufträge*
   - *Operative Fertigungsplanung*

Anwenderlösung für die Bürokommunikation

Am IIR wurden auf der Basis von ROLANET 1 erste Anwendungslösungen für die Bürokommunikation erprobt und für den experimentellen LAN-Betrieb folgende Möglichkeiten genutzt:

- Bedienerkommunikation
- Virtuelles Diskettensystem (Filedienst).

Die Bedienerkommunikation gestattet im LAN die Übermittlung von kurzen Mitteilungen, Rundsprüchen und Anfragen direkt auf die Bildschirme der angeschlossenen Stationen.
Mit dem *Virtuellen Diskettensystem* können durch die Teilnehmer des LAN entfernte Dateien für die Verarbeitung genutzt werden. Eine Station (A 5120) wurde als Dateiserver ausgewählt. Die an dieser Station verfügbaren 3 Diskettenlaufwerke sind für den Austausch von Dateien vorgesehen (Bild 2).
Für die Nutzung des Virtuellen Diskettensystems für die Bürokommunikation wurden Kommunikations- und Anwendungsdateien eingeführt.
Kommunikationsdateien sind Dateien, die von der LAN-Station *Sender* zum Dateiserver übertragen werden und dort von den LAN-Stationen *Empfänger* gelesen werden können.
Der Empfänger ist für die Übernahme und damit für das Löschen der Datei verantwortlich. Eine Sonderform der Kommunikationsdateien sind entsprechend gekennzeichnete Rundschreiben, die nur vom Sender gelöscht werden dürfen.
Anwendungsdateien sind Dateien, die auf dem Dateiserver bereitgestellt werden und

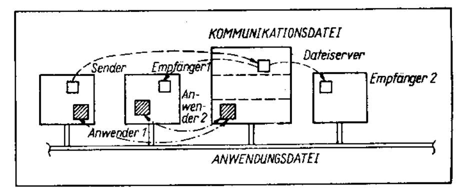


Bild 2 Arbeitsweise mit dem virtuellen Diskettensystem

dort allen LAN-Stationen zur Verfügung stehen. Der Sender ist verantwortlich für die Pflege dieser Dateien. Als Anwendungsdateien wurden durch REDABAS nutzbare Datenbankfiles auf dem Dateiserver bereitgestellt. Diese Files können von jeder Station aus mit Hilfe des lokal verfügbaren Datenbankbetriebssystems für Abfragen genutzt werden.
Die Pflege der Files ist zentral geregelt. Darüber hinaus wurden als Anwendungsdateien auch Textdateien bereitgestellt. Diese Dateien enthalten oft im Institut benötigte Texte, wie Vertragsrahmen, Anweisungen, Auszeichnungsformulare u. ä.
Auf der Grundlage dieser Dateitypen wurden Festlegungen getroffen, über den Dateinamen und den Dateityp eine eindeutige Kennzeichnung von Sender und Empfänger zu erreichen.
Die Nutzung des virtuellen Diskettensystems erfolgt für diese Kommunikationsformen ausschließlich mit Hilfe des Textprozessors TP. Dadurch wurde erreicht, daß die Nutzer dieser Anwendungsformen, die Sekretärinnen und Sachbearbeiter, keine neuen, LAN-spezifischen Handlungen ausführen müssen.

Fortsetzung auf S. 200

Der Floppy-Disk-Controller U 8272 D und sein Einsatz (Teil 2)

Dr. Eberhard Böhl
VEB Forschungszentrum Mikroelektronik Dresden

Beschreibung der Befehle

Der U 8272 kann 15 verschiedene Befehle ausführen, von denen jeder aus 3 Phasen besteht:

Befehlsphase:
Eingabe mehrerer Bytes Informationen zur genauen Spezifizierung eines Befehls

Ausführungsphase:
Durchführung der angewiesenen Operationen

Ergebnisphase:
Übermittlung von Status- und Adreßfeldinformationen, die über die ordnungsgemäße Abarbeitung des Befehls Auskunft geben.

In der Befehls- und Ergebnisphase muß vor jedem Schreiben bzw. Lesen eines Bytes das Statusregister gelesen werden. In der Ausführungsphase ist das nicht erforderlich. Bei Lese- und Schreibbefehlen wird im Nicht-DMA-Modus über das INT-Signal und im DMA-Modus über das DRQ-Signal die Bereitschaft zum Datenaustausch signalisiert. Sobald der Datenaustausch erfolgt ist bzw. die DMA-Bestätigung ($\overline{DAK} = 0$) aktiv wird, werden die Anforderungssignale zurückgesetzt. Nach Abarbeitung aller Daten im DMA-Modus wird mit dem TC-Signal ein INT-Signal gebildet, das den Beginn der Ergebnisphase signalisiert. Dieses INT-Signal wird zurückgesetzt, sobald das erste Datenbyte der Ergebnisphase gelesen wurde. Es ist unbedingt erforderlich, alle Datenbytes in der Ergebnisphase zu lesen, da der U 8272 vorher keinen neuen Befehl akzeptiert. Die als ST0, ST1, ST2 bzw. ST3 ausgewiesenen Statusregister sind nur in der Ergebnisphase verfügbar und nur, wenn dies in der Befehlsspezifikation angegeben wird. Im Gegensatz zum Hauptstatusregister, das immer gelesen werden kann, sind die Statusregister in der Ergebnisphase wie Datenregister (mit $A_0 = 1$) zu lesen.
Nachdem alle Bytes zur Befehlsspezifikation in der Befehlsphase eingegeben wurden, beginnt automatisch die Ausführungsphase, die beendet wird, wenn das letzte Datenbyte übertragen wurde. Ein Schreib-/Lesebefehl kann in der Ausführungsphase durch Anlegen des TC-Signales abgebrochen werden. Nach dem Lesen aller Bytes der Ergebnisphase wird der Befehl abgeschlossen und der U 8272 ist bereit zur Eingabe eines neuen Befehls.

Befehlsspezifikation

Die 15 Befehle des U 8272 sind:

READ DATA (Daten lesen)
READ DELETED DATA (Lesen irgnorierter Daten)
WRITE DATA (Daten schreiben)
WRITE DELETED DATA (Schreiben ignorierter Daten)
READ A TRACK (Lesen einer Spur)
READ ID (Lesen des Identifikationsfeldes)
FORMAT A TRACK (Formatisieren einer Spur)
SCAN EQUAL (Testen auf Gleichheit)
SCAN LOW OR EQUAL (Testen auf kleiner oder gleich)
SCAN HIGH OR EQUAL (Testen auf größer oder gleich)
RECALIBRATE (Rücksetzen)
SENSE INTERRUPT STATUS (Abfragen des Interruptstatus)
SPECIFY (Wertzuweisung)
SENSE DRIVE STATUS (Abfragen des Laufwerkstatus)
SEEK (Suche).

Zur ordnungsgemäßen Durchführung eines solchen Befehls ist ein Informationstausch von mehreren Bytes zwischen der CPU und dem U 8272 erforderlich. Für einige Befehle ist der spezielle Ablauf in Tafel 3 bzw. Tafel 4 dargestellt. Die dabei benutzten Abkürzungen sind in Tafel 5 erläutert. In Tafel 6 werden die einzelnen Bits des Hauptstatusregisters erläutert, und in Tafel 7 sind die Statusregister erklärt, die nur in der Ergebnisphase eines Befehls verfügbar sind.

READ DATA

Wie in Tafel 3 gezeigt, ist die Eingabe von 9 Bytes in den U 8272 notwendig, um ihm die notwendige Information zur Ausführung eines Daten-Lese-Befehls zu übermitteln.
Zu Beginn der Ausführungsphase veranlaßt der U 8272 das Kopfladen (sofern der Kopf nicht schon geladen ist), und nach dem Abwarten mit der vorgeschriebenen Kopfladezeit (im SPECIFY-Befehl vorher festgelegt) beginnt das Lesen der ID-Adressmarken und ID-Felder. Sobald die im Befehl angegebene Sektornummer (R), gespeichert im ID-Register (IDR), mit der von der Diskette gelesenen Sektornummer übereinstimmt, gibt der U 8272 die Daten des Datenfeldes byteweise über den Datenbus aus. Im Nicht-DMA-Modus erzeugt der U 8272 bei jedem ausgabebereiten Datenbyte einen Interrupt (INT = 1), der in der vorgeschriebenen Zeit (bei MFM alle 13 μs) behandelt werden muß. Falls eine so schnelle Interrupt-Behandlung nicht möglich ist, kann auch durch Abfrage des Hauptstatusregisters (RQM) die Information erhalten werden, ob ein Datenbyte zu lesen ist. Im DMA-Modus wird das Signal DRQ anstatt INT aktiviert. Die Unterscheidung zwischen DMA-Modus und Nicht-DMA-Modus erfolgt im SPECIFY-Befehl durch das Bit ND (Non-DMA).
Ist die Leseoperation eines Sektors beendet, wird die Sektornummer um 1 erhöht, und die Daten des nächsten Sektors werden gelesen und ausgegeben (Multi-Sector-Read-Operation). Der Daten-Lese-Befehl wird durch Anlegen des Signals TC = 1 an den U 8272 beendet. Der U 8272 beendet damit sofort seine Datenausgabe zum Prozessor, liest aber noch die restlichen Daten des laufenden Sektors, um die CRC-Bytes zu prüfen.
Bei Angabe von MT = 1 im Befehlscode liest der U 8272 die Daten von beiden Seiten der Diskette (Multi-Track), beginnend von Seite 0. Es sind dabei nur die zwei Spuren des gleichen Zylinders durch diesen Befehl nacheinander lesbar.
Ist N = 0, so gibt DTL die Datenlänge an, die der U 8272 als einen Sektor behandeln soll (nur bei einfacher Datendichte). Ist DTL kleiner als 128, so werden nur entsprechend weniger Datenbytes auf den Bus zum Prozessor gegeben, obwohl der U 8272 alle Daten des Sektors liest und zur korrekten CRC-Berechnung verarbeitet. In Abhängigkeit von der Befehlsbedingung kann der U 8272 eine Mehrfachsektor-Leseoperation ausführen. Für

Tafel 3 Lesebefehl U 8272

READ DATA

Phase	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Bem.
Befehl	W	MT	MFM	SK	0	0	1	1	0	Code
		X	X	X	X	X	HDS	DS1	DS0	
					C					
					H					Sektor-
					R					Adr.
					N					(ID)
					EOT					
					GPL					
	W				DTL					
Ausf.		.	.	.	.	.	.	.	.	Daten
Ergebnis	R				ST0					
					ST1					Status
					ST2					
					C					
					H					
					R					
	R				N					

Tafel 4 Befehle des U 8272

Phase	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Bem.
	RECALIBRATE									
Bef.	W	0	0	0	0	0	1	1	1	
	W	0	0	0	0	0	0	DS1	DS0	
	SPECIFY									
Bef.	W	0	0	0	0	0	0	1	1	
	W		SRT				HUT			
	W		HLT						ND	
	SEEK									
Bef.	W	0	0	0	0	1	1	1	1	
	W	0	0	0	0	0	HDS	DS1	DS0	
	W				NCN					
	SENSE INTERRUPT STATUS									
Bef.	W	0	0	0	0	1	0	0	0	
Erg.	R				ST0					
	R				PCN					

Literatur
/1/ Merkel, G.: Kommunikationstechnologien heute und morgen. Spektrum 18 (1987) 10, S. 20
/2/ Jack, N.: Symposium „Das lokale Rechnernetz ROLANET“. NTB 31 (1987) 5, S. 139

✉ KONTAKT ☎
Zentrum für wissenschaftlichen Gerätebau der AdW der DDR, Rudower Chaussee 6, Berlin, 1199; Tel. 6743382 (Dr. Grubba)

Fortsetzung von S. 199

Tafel 5 Erläuterung der Befehlsmnemonik

Symbol	Name	Beschreibung
C	Zylindernummer	ausgewählte Spur der Diskette (0 bis 76)
D7–D0	Datenbus	8-Bit-Datenbus
DTL	Datenlänge	wenn N = 0 ist, gibt DTL die Datenlänge/Sektor ≤ 128 Byte an (nur bei FM)
DS0, DS1	Laufwerksanwahl	ausgewählte Laufwerksnummer
EOT	Spurende	letzte Sektornummer eines Zylinders
GPL	Lückenlänge	Länge der Lücke 3 zwischen den Sektoren
H	Kopf	Kopfnummer 0 oder 1 entsprechend ID-Feld
HDS	Kopfauswahl	ausgewählte Kopfnummer (0 oder 1)
HLT	Kopfladezeit	programmierbar von 2 bis 254 ms
HUT	Kopfentladezeit	programmierbar von 16 bis 240 ms
MFM	FM oder MFM	FM bei MFM = 0, MFM bei MFM = 1
MT	Mehrfachspur	bei MT = 1 werden beide Spuren eines Zylinders bearbeitet
N	Zahl der Datenbytes pro Sektor	Wert = 128 Bytes · 2^N
NCN	neue Zylindernummer	gewünschte Kopfposition
ND	Nicht-DMA-Modus	für Operation im Nicht-DMA-Modus: ND = 1
PCN	aktuelle Zylindernummer	aktuelle Kopfposition
R	Aufzeichnung	Sektornummer des zu bearbeitenden Sektors
R/W	Lesen/Schreiben	Lesesignal (R) bzw. Schreibsignal (W)
SK	Sprung	Überspringen ignorierter Daten
SRT	Schrittimpulsrate	programmierbar von 1 bis 16 ms
STO...3	Statusregister	Statusinformation nach Befehlsausführung (nur in Ergebnisphase verfügbar). Diese Statusinformation ist über den Datenpuffer zu lesen (mit A_0 = 1, wie ein Datenregister).
STP	Schritt	Während SCAN-Operation werden die Daten in mehreren Sektoren byteweise verglichen. Bei STP = 1 fortlaufend, bei STP = 2 jeder 2. Sektor

Tafel 6 Erläuterung der Bits des Hauptstatusregisters

Bit-Nr.	Name	Symbol	Beschreibung
D_0	FDD0 busy	D_0B	FDD-Nummer 0 ist im Such-Modus
D_1	FDD 1 busy	D_1B	FDD-Nummer 1 ist im Such-Modus
D_2	FDD 2 busy	D_2B	FDD-Nummer 2 ist im Such-Modus
D_3	FDD 3 busy	D_3B	FDD-Nummer 3 ist im Such-Modus
D_4	FDC busy	CB	Ausführung eines Lese- oder Schreibbefehls
D_5	Nicht-DMA-Modus	NDM	Der FDC arbeitet im Nicht-DMA-Modus. Dieses Bit wird nur während der Ausführungsphase in den Nicht-DMA-Modus gesetzt. Übergang zum 0-Zustand zeigt an, daß die Ausführungsphase beendet wurde.
D_6	Data Input/Output	DIO	Gibt die Richtung des Datentransfers zwischen dem FDC und dem Prozessor an: Falls DIO = 1: Daten vom Datenregister (FDC) zum Prozessor, falls DIO = 0: Daten vom Prozessor zum Datenregister des FDC.
D_7	Request for Master	RQM	Datenregister meldet Bereitschaft zum Datenaustausch (Senden und Empfangen von Daten zu oder vom Prozessor). Die Bits DIO und RQM sollten zur Realisierung des Handshake-Prinzips mit dem Prozessor benutzt werden (mit der Bedeutung als Bereitschafts- und Richtungssignal).

N = 0 hat DTL keine Bedeutung und sollte FF gesetzt werden. Nach Beendigung des Daten-Lese-Befehls bleibt der Kopf noch geladen, bis die Kopfentladezeit (HUT) verstrichen ist (im SPECIFY-Befehl festgelegt). Die unmittelbare Ausführung eines Schreib-Lese-Befehls vor Ablauf der Kopfentladezeit ist möglich und erspart ein erneutes Laden des Kopfes. Falls ein Befehl nicht ordentlich beendet werden kann, weil z. B. der U 8272 den richtigen Sektor nach zweimaligem Erkennen des Indexloches nicht gefunden hatte oder die CRC-Prüfung einen falschen Wert ergab, so werden im STO die Bits 7 und 6 auf 0 bzw. 1 gesetzt und außerdem in den anderen Statusregistern entsprechende Bits je nach Fehlerart gesetzt.

Nach der Beendigung eines Lese-Schreib-Befehls durch den Prozessor sendet der U 8272 in der Ergebnisphase das ID-Feld des nächstfolgenden Sektors (nach dem zuletzt ausgegebenen Sektor).

WRITE DATA

Auch beim Schreibbefehl sind zur Spezifikation 9 Datenbytes an den U 8272 zu übertragen. Dabei unterscheidet sich nur das erste Byte (der Befehlscode) von denen beim Befehl Datenlesen:

MT MFM 0 0 0 1 0 1 Befehlscode

Die 3 folgenden Bytes sind wie bei READ DATA (Tafel 3) anzufügen. Der Befehlsablauf ist äquivalent zu dem Lesebefehl, nur werden nach dem Lesen des Identifikationsfeldes bei Übereinstimmung der Sektornummer mit der vorgegebenen die Daten byteweise vom Prozessor übernommen und in das Datenfeld geschrieben. Ist ein Sektor vollständig mit Daten beschrieben, so wird die Sektornummer um 1 erhöht und der Schreibvorgang im nächsten Sektor fortgeführt. Dieser Schreibvorgang wird weitergeführt bis zum Vorliegen des Terminal-Count-Signals (TC = 1); in diesem Fall werden keine Daten vom Prozessor mehr angefordert. Falls ein Datenfeld zum Zeitpunkt TC = 1 noch nicht vollständig abgearbeitet war, so werden die restlichen Daten dieses Datenfeldes mit 00H aufgefüllt. Die Ergebnisphase ist wie bei READ DATA (Tafel 3).

WRITE DELETED DATA

Gleicht dem Befehl WRITE DATA, nur wird die normale Daten-Adreßmarke zu Beginn des Datenfeldes durch eine ignorierte Daten-Adreßmarke ersetzt. Der Befehlscode lautet:

MT MFM 0 0 1 0 0 1 Befehlscode;

anschließend sind die 8 Bytes wie bei READ DATA (Tafel 3) zu übertragen.

READ DELETED DATA

Wie READ DATA, nur falls (bei SK = 0) der U 8272 eine normale Daten-Adreßmarke zu Beginn eines Datenfeldes erkennt, liest er alle Daten des Sektors und bricht dann den Befehl mit einer Fehlermeldung (CM = 1 in ST2) ab. Bei SK = 1 überspringt der U 8272 den Sektor mit normaler Daten-Adreßmarke und liest den nächsten Sektor. Der Befehlscode lautet:

MT MFM SK 0 1 1 0 0 Befehlscode;

die restlichen 8 Bytes entsprechen denen von READ DATA (Tafel 3).

READ A TRACK

Unmittelbar nach dem Erkennen des Indexloches werden alle Daten einer Spur als fortlaufender Datenblock gelesen. Findet der U 8272 einen Fehler in den ID- oder Daten-CRC-Prüfbytes, so wird das Lesen fortgesetzt. Der U 8272 vergleicht die ID-Informationen, die aus jedem Sektor gelesen werden, mit den im IDR gespeicherten Informationen.

Falls keine Übereinstimmung vorliegt, wird das ND-Flag im Stausregister 1 auf 1 gesetzt. Mehrfachspur- und Sprungoperationen sind bei diesem Befehl nicht möglich. Der Befehl ist wie folgt einzugeben:

0 MFM SK 0 0 0 1 0 Befehlscode;

die restlichen 8 Bytes wie bei READ DATA (Tafel 3).

READ ID

Mit dem Befehl

0 MFM 0 0 1 0 1 0 Befehlscode

0 0 0 0 0 HDS DS1 DS0

überträgt der U 8272 die Werte des ersten lesbaren ID-Feldes in ein internes Register und gibt sie in der Ergebnisphase (wie bei READ DATA) an den Prozessor weiter.

FORMAT A TRACK

Mit dem Befehl

0	MFM	0	0	1	1	0	1
0	0	0	0	0	HDS	DS1	DS0

______________ N ______________

______________ SC ______________

______________ GPL ______________

______________ D ______________

wird eine Spur mit (N + 1) · 128 Byte pro Sektor, SC Sektoren, der Lückenlänge GPL und den Daten D beschrieben. Das ID-Feld eines jeden Sektors wird durch den Prozessor vorgegeben, indem in der Ausführungsphase des Befehls jeweils 4 Datenbytes für jeden Sektor abgefordert werden: C, H, R, N. Dadurch besteht die Möglichkeit, eine Diskette auch mit ungeordneter Reihenfolge der Sektornummern zu formatieren. In der Ergebnisphase (wie bei READ DATA) ergibt sich für R gegenüber dem eingegebenen Wert eine Erhöhung um 1, da der U 8272 weiterzählt.

SCAN-Befehle

SCAN EQUAL:

MT MFM SK 1 0 0 0 1 Befehlscode

SCAN LOW OR EQUAL:

MT MFM SK 1 1 0 0 1 Befehlscode

SCAN HIGH OR EQUAL:

MT MFM SK 1 1 1 0 1 Befehlscode

Für alle 3 Testbefehle sind nach dem Befehlscode die restlichen 8 Bytes wie bei READ DATA einzugeben. Dabei bedeutet das letzte Byte aber nicht DTL (die Datenlänge), sondern STP (Schrittweite). Bei STP = 1 wird jeder Sektor geprüft, bei STP = 2 jeder zweite, usw.

Der U 8272 vergleicht bei allen diesen Befehlen, ob die Bedingung

$D_{FDD} = D_{PROZESSOR}$, $D_{FDD} \leqq D_{PROZESSOR}$ bzw. $D_{FDD} \geqq D_{PROZESSOR}$

erfüllt ist. Dabei kann das Datenbyte FF vom Prozessor als Maske benutzt werden, das immer die Vergleichsbedingung erfüllt. Ist nach dem Vergleich eines gesamten Sektors

Tafel 7 Erläuterung der Statusregister

Nr.	Bit Name	Symbol	Beschreibung
STATUSREGISTER 0			
D_7 D_6	Interrupt Code	IC	$D_7 = 0$ und $D_6 = 0$ Normale Beendigung eines Befehls (Normal Termination – NT). Der Befehl wurde vollständig und exakt abgearbeitet.
			$D_7 = 0$ und $D_6 = 1$ Unnormale Beendigung eines Befehls (Abnormal Termination – AT). Die Ausführung eines Befehls wurde begonnen, aber nicht erfolgreich abgeschlossen.
			$D_7 = 1$ und $D_6 = 0$ Fehlerhafte Befehlsausgabe (Invalid Command Issue – IC). Der ausgesendete Befehl wurde nicht gestartet.
			$D_7 = 1$ und $D_6 = 1$ Unnormale Beendigung wegen Pegelwechsel auf der READY-Leitung vom FDD während der Befehlsausführung.
D_5	SEEK End	SE	Wenn der FDC den Suchbefehl abschließt, wird dieses Flag auf High (1) gesetzt.
D_4	Equipment Check	EC	Dieses Flag wird gesetzt, wenn vom FDD ein Fehlersignal empfangen wird (FAULT) oder wenn beim Rücksetzbefehl nach 77 Schrittimpulsen noch kein Spur-0-Signal gemeldet wurde.
D_3	Not Ready	NR	Dieses Flag wird gesetzt, wenn ein Lese- oder Schreibbefehl ausgesendet wurde und der FDD im Not-Ready-Zustand ist. Das Flag wird auch gesetzt, wenn ein Lese- oder Schreibbefehl für Seite 1 einer einseitigen Diskettenstation ausgesendet wurde.
D_2	Head Address	HD	Dieses Flag wird genutzt, um bei einem Interrupt die Nummer des Kopfes zu melden.
D_1	Drive Select 1	DS 1	Diese Flags werden genutzt, um bei einem Interrupt die Treibernummer auszugeben.
D_0	Drive Select 0	DS 0	
STATUSREGISTER 1			
D_7	End of Cylinder	EN	Dieses Flag wird gesetzt, wenn der FDC versucht, zu einem Sektor hinter dem letzten Sektor eines Zylinders zuzugreifen.
D_6			Nicht verwendet. Dieses Bit ist immer low (0).
D_5	Data Error	DE	Wenn der FDC einen CRC-Fehler im ID-Feld oder im Datenfeld findet, wird dieses Flag gesetzt.
D_4	Over Run	OR	Dieses Flag wird gesetzt, wenn der FDC während eines Datentransfers vom Zentralsystem nicht während einer bestimmten Zeit bedient wird.
D_3			Nicht verwendet. Dieses Bit ist immer low (0).
D_2	No Data	ND	Das Flag wird gesetzt, wenn der FDC während der Ausführung der Befehle READ DATA, READ DELETED DATA oder SCAN nicht den Sektor finden kann, der im ID-Register vorgegeben ist.
			Das Flag wird gesetzt, wenn der FDC während der Ausführung des Befehls READ ID nicht das ID-Feld fehlerfrei lesen kann.
			Das Flag wird gesetzt, wenn vom FDC bei der Ausführung des Befehls „Lesen eines Zylinders" der Startsektor nicht gefunden wird.
D_1	Not Writetable	NM	Das Flag wird gesetzt, wenn der FDC während der Ausführung der Befehle WRITE DATA, WRITE DELETED DATA und FORMAT A TRACK ein Schreibschutzsignal registriert.
D_0	Missing Address Mark	MA	Wenn der FDC bis zum zweiten Passieren des Indexloches keine ID-Adreßmarke gefunden hat, wird das Flag gesetzt.
			Wenn der FDC die Daten-Adreßmarke oder die ignorierte Daten-Adreßmarke nicht finden kann, wird dieses Flag ebenfalls gesetzt (gleichzeitig mit dem MD-Flag [Missing Address Mark in Data Field] des Statusregisters 2).
STATUSREGISTER 2			
D_7			Nicht verwendet. Dieses Bit ist immer low (0).
D_6	Control Mark	CM	Dieses Flag wird gesetzt, wenn der FDC während der Ausführung eines Datenlese- oder Testbefehls einen Sektor mit ignorierter Daten-Adreßmarke erreicht.
D_5	Data Error in Data Field	DD	Das Flag wird gesetzt, wenn der FDC einen CRC-Fehler im Datenfeld erkennt.
D_4	Wrong Cylinder	WC	Dieses Bit ist in Verbindung mit dem ND-Flag zu betrachten. Das WC-Flag wird gesetzt, wenn der Inhalt von C der Diskettenspur von dem im Befehl angegebenen C-Wert abweicht.
D_3	Scan Equal Hit	SH	Das Flag wird gesetzt, wenn während der Ausführung des Testbefehls die Bedingung GLEICH erfüllt wird.
D_2	Scan Not Satisfied	SN	Das Flag wird gesetzt, wenn der FDC während der Ausführung des Testbefehls keinen Sektor findet, der die Testbedingung erfüllt.
D_1	Bad Cylinder	BC	Dieses Bit ist in Verbindung mit dem ND-Flag zu betrachten. Das Flag wird gesetzt, wenn der Inhalt von C der Diskette von dem im Befehl angegebenen C-Wert abweicht und der Inhalt von $C = FF_{Hex}$ ist.
D_0	Missing Address Mark in Data Field	MD	Das Flag wird gesetzt, wenn der FDC beim Auslesen von Daten von der Diskette keine Daten-Adreßmarke oder keine ignorierte Daten-Adreßmarke finden kann.
STATUSREGISTER 3			
D_7	Fault	FT	Dieses Bit wird benutzt, um den Status des Fehlersignals vom FDD anzuzeigen.
D_6	Write Protected	WP	Dieses Bit wird benutzt, um den Schreibschutz vom FDD zu melden.
D_5	Ready	RDY	Dieses Bit wird benutzt, um die Bereitschaft vom FDD zu melden.
D_4	Track 0	T0	Dieses Bit wird benutzt, um vom FDD das Spur-0-Signal anzuzeigen.
D_3	Two Sides	TS	Dieses Bit wird benutzt, um vom FDD das Signal Zweiseitenbetrieb zu melden.
D_2	Head Address	HD	Dieses Bit zeigt den Zustand des Seitenwahlsignals (Side Select) zum FDD an.
D_1	Drive Select	DS 1	Diese Bits zeigen den Zustand der Stationswahlsignale (Drive Select) an.
D_0	Drive Select 0	DS 0	

die Testbedingung nicht erfüllt, so wird die Sektornummer um STP erhöht und der Vergleich fortgesetzt.

SEEK

Durch den SEEK-Befehl (Suchbefehl) wird der Lese-/Schreibkopf des Laufwerkes gezielt von Zylinder zu Zylinder bewegt. Dabei wird die aktuelle Kopfposition (PCN) ständig mit der gewünschten Kopfposition (NCN) verglichen. Im Ergebnis des Vergleiches PCN < NCN bzw. PCN > NCN werden das Richtungssignal (DR) auf High (1) bzw. Low (0) gesetzt und die Schrittimpulse ausgesendet. Die Folgefrequenz der Schrittimpulse ist vorher im SPECIFY-Befehl durch den Wert von SRT (Stepping Rate Time) festgelegt worden. Mit PCN = NCN wird das Aussenden der Stepimpulse abgebrochen und der Befehl beendet. Während der Ausführungsphase eines Suchbefehls kann ein weiterer Suchbefehl für ein anderes Laufwerk gestartet werden – das ist für maximal 4 Laufwerke gleichzeitig möglich.

Es ist zu beachten, daß bei einem Lese-/Schreibbefehl keine Suchbefehle ausgeführt werden. Deshalb sollte einem Lese-/Schreibbefehl vorausgehen:

1. ein Suchbefehl
2. Sense Interrupt Status
3. Red ID.

Der PCN-Zähler ist nur durch den Rücksetzbefehl (Recalibrate) zu eichen. Der Suchbefehl ist in Tafel 4 dargestellt.

RECALIBRATE

Durch den Rücksetzbefehl wird der Lese-/Schreibkopf des Laufwerkes auf die Spur 0 zurückgesetzt und der Inhalt des PCN-Zählers gelöscht. Der U 8272 sendet so lange Schrittimpulse in Verbindung mit dem Richtungssignal DR = 1 aus, bis das Spur-0-Signal (T0) high wird. Damit wird der Befehl beendet. Ist nach dem Aussenden von 77 Schrittimpulsen immer noch T0 = 0, so wird der Befehl abgebrochen. Alle vier Laufwerke können parallel rückgesetzt werden. Der Recalibrate-Befehl ist in Tafel 4 dargestellt.

SENSE INTERRUPT STATUS

Ein Interrupt-Signal wird vom U 8272 in folgenden Fällen erzeugt:
1. Bei Erreichen der Ergebnisphase folgender Befehle:
- Datenlesen
- Zylinderlesen
- Lesen der Identifikation
- Lesen ignorierter Daten
- Datenschreiben
- Formatieren eines Zylinders
- Schreiben ignorierter Daten
- Testbefehl

2. bei Pegelwechsel auf den Ready-Leitungen der Laufwerke
3. nach Beendigung eines Such- oder Rücksetzbefehls
4. während der Ausführungsphase im Nicht-DMA-Betrieb.

Die Ursachen 1. und 4. werden vom Prozessor erwartet und sind leicht zu erkennen; die Ursachen 2. und 3. sind jedoch nur mittels des Befehls SENSE INTERRUPT STATUS entschlüsselbar. Durch diesen Befehl wird das Interrupt-Signal zurückgesetzt und dessen Ursache über die Bits 5, 6 und 7 des Statusregisters 0 abgefragt. Da weder Suchnoch Rücksetzbefehl eine Ergebnisphase besitzen, ist mit diesem Befehl die Kontrolle für die exakte Durchführung und fehlerfreie Beendigung möglich.
Der Sense-Interrupt-Befehl ist in Tafel 4 dargestellt.

SPECIFY

Durch diesen Befehl werden den 3 internen Zeitgebern Werte zugeordnet:
Kopfentladezeit (HUT):
1 ≙ 16 ms, 2 ≙ 32 ms, ..., F ≙ 240 ms
Schrittimpulsrate (SRT):
F ≙ 1 ms, E ≙ 2 ms, ..., 0 ≙ 16 ms
Kopfladezeit (HLT):
01 ≙ 2 ms, 02 ≙ 4 ms, ..., FE ≙ 254 ms
Die angegebenen Werte gelten für eine Taktfrequenz von 8 MHz. Bei 4 MHz verdoppeln sich die Zeiten bei gleicher Wertzuweisung im Befehl Specify.
Die Schrittrate sollte 1 ms länger als die minimale vom FDD geforderte Zeit programmiert werden. Durch das ND-Bit (Non DMA) wird zwischen Nicht-DMA-Betrieb (ND = 1) und DMA-Betrieb (ND = 0) unterschieden. Der Specify-Befehl ist in Tafel 4 dargestellt.

SENSE DRIVE STATUS

Dieser Befehl wird benutzt, um den Status eines Laufwerkes abzufragen. Er hat folgende Form:

0	0	0	0	0	1	0	0
0	0	0	0	0	HDS	DS1	DS0

———————— ST3 ————————

In ST3 ist die Statusinformation des angewählten Laufwerkes enthalten.

INVALID

Alle Befehle, die nicht einem vorangegangenen Befehl entsprechen, sind ungültig. Der U 8272 beantwortet einen solchen ungültigen Befehlscode mit der Ausgabe des Statusregisters 0, das den Wert 80 H enthält. Das Erreichen der Ergebnisphase wird in diesem Fall nicht durch ein INT-Signal angezeigt, sondern es ist das Hauptstatusregister zu lesen. Nachfolgend ist ein solcher Befehl dargestellt:

———————— ungültiger Code ————————

Befehlscode

———————— ST0 ————————

Ergebnisphase
Jeder solcher Befehl kann als NOP-Befehl verwendet werden.

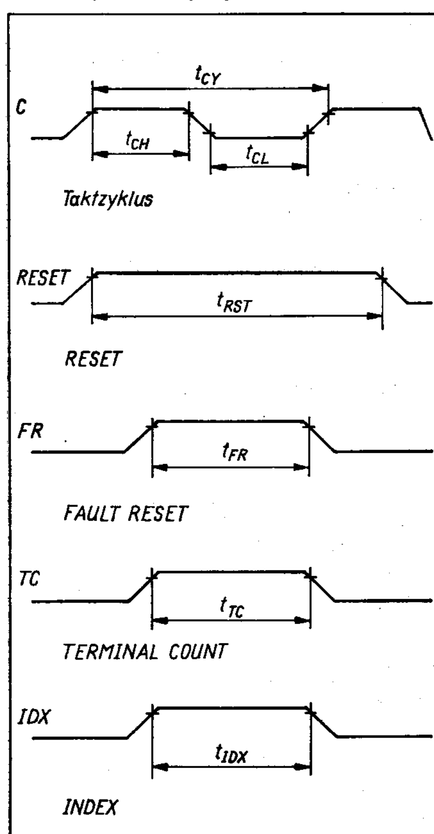
Grenzwerte, Betriebsbedingungen und Kennwerte

Der U 8272 arbeitet bei einer Betriebsspannung von 4,75 bis 5,25 V bei 0 bis 70 °C Umgebungstemperatur. Die Eingangspegel betragen −0,8 V (low) bzw. −2,0 V (high), für die Takteingänge −0,65 V bzw. 2,4 V. Der Typ U 8272 D08 arbeitet mit einer minimalen Taktperiode von 125 ns bei minimal 40 ns Highphase und minimal 50 ns Lowphase. Für den Typ U 8272 D04 verdoppeln sich diese Werte. Mit dem Beginn des Aktivwerdens des $\overline{RD}$- bzw. $\overline{WR}$-Eingangs müssen A_0, $\overline{CS}$ und $\overline{DAK}$ gültig sein.
An den Ausgängen liefert der U 8272 einen Lowpegel von 0,45 V bei 2 mA bzw. einen Highpegel von 2,4 V bei −200 μA Belastung. Die Leckströme der Ausgänge liegen unter 10 μA.
Ausführlichere Angaben hierzu enthalten die Tafeln 8, 9 und 10 und die dazugehörigen Bilder 6 bis 12.

***Tafel 8* Grenzwerte**

Kenngröße	Symbol	Einheit	min.	max.
Betriebsspannung	U_{CC}	V	−0,5	7,0
Eingangsspannung	U_I	V	−0,5	7,0
Ausgangsspannung	U_O	V	−0,5	7,0
Verlustleistung	P_V	W	–	1,5
Umgebungstemperatur bei Belastung	ϑ_a	°C	0	70
Lagertemperatur	ϑ_s	°C	−40	125

***Bild 6* Impulsbedingungen**



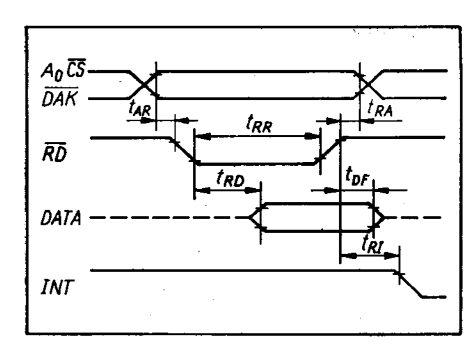


***Bild 7* Prozessor-Leseoperation**

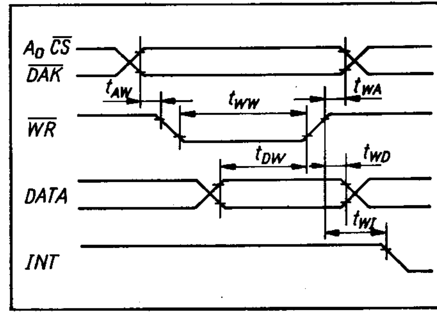


***Bild 8* Prozessor-Schreiboperation**

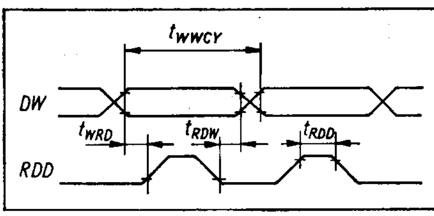


***Bild 9* FDD-Leseoperation**

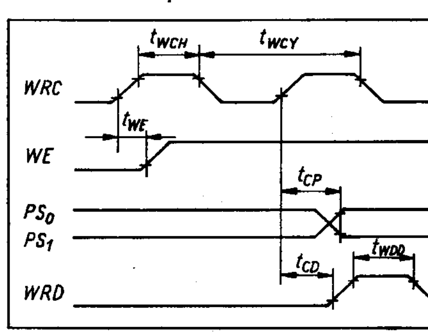


***Bild 10* FDD-Schreiboperation**

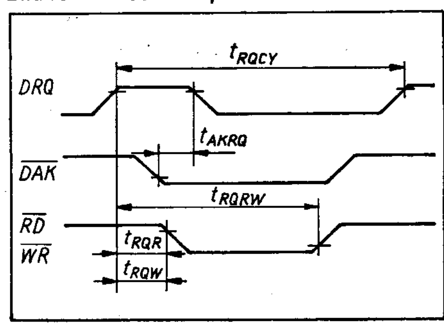
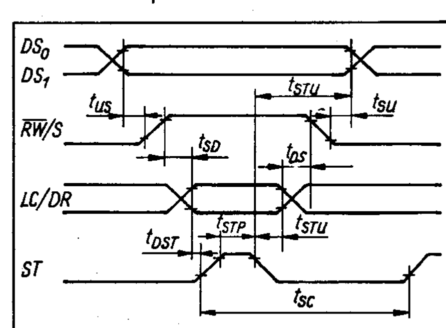


***Bild 11* DMA-Operation**
***Bild 12* Such-Operation**



Tafel 9 Betriebsbedingungen

Kenngröße	Symbol	Einheit	min.	typ	max.	Bedingung
Umgebungstemperatur	ϑ_a	°C	0	–	70	
Betriebsspannung	U_{CC}	V	4,75	–	5,25	
L-Eingangsspannung	U_{IL}	V	−0,5	–	0,8	
H-Eingangsspannung	U_{IH}	V	2,0	–	$U_{CC}+0{,}5$	
L-Eingangsspannung für CLK und WRC	$U_{IL}(C)$	V	−0,5	–	0,65	
H-Eingangsspannung für CLK und WRC	$U_{IH}(C)$	V	2,4	–	$U_{CC}+0{,}5$	
Taktperiode	t_{CY}	ns	125	–	500	x[1]
H-Taktperiode	t_{CH}	ns	40	–	–	x[1]
L-Taktperiode	t_{CL}	ns	50	–	–	x[1]
Anwahl bis $\overline{RD}$↓	t_{AR}	ns	0	–	–	
$\overline{RD}$↑ bis Anwahl	t_{RA}	ns	0	–	–	
$\overline{RD}$↓ bis $\overline{RD}$↑	t_{RR}	ns	250	–	–	
Anwahl bis $\overline{WR}$↓	t_{AW}	ns	0	–	–	
$\overline{WR}$↑ bis Anwahl	t_{WA}	ns	0	–	–	
$\overline{WR}$↓ bis $\overline{WR}$↑	t_{WW}	ns	250	–	–	
Daten bis $\overline{WR}$↑	t_{DW}	ns	150	–	–	
$\overline{WR}$↑ bis Daten	t_{WD}	ns	5	–	–	
TC↑ bis TC↓	t_{TC}	t_{CY}	1	–	–	
RESET↑ bis RESET↓	t_{RST}	t_{CY}	14	–	–	
IDX↑ bis IDX↓	t_{IDX}	t_{CY}	–	10	–	
WRC↑ bis WRC↓	t_{WCH}	ns	100	250	350	x[2]
Einschaltflanke WRC	t_{WCR}	ns	–	–	20	
Ausschaltflanke WRC	t_{WCF}	ns	–	–	20	
RDD↑ bis RDD↓	t_{RDD}	ns	40	–	–	
RDD↓ bis DW	t_{RDW}	ns	15	–	–	
DW bis RDD↑	t_{WRD}	ns	15	–	–	
WRC↑ bis WRC↑	t_{WCY}	µs	–	2	–	FM 8"
			–	1	–	MFM 8"
WRC↑ bis WRC↑	t_{WCY}	µs	–	4	–	FM 5¼"
			–	2	–	MFM 5¼"
DW↑ bis DW↓	t_{WWCY}	µs	–	2	–	FM 8"
DW↓ bis DW↑			–	1	–	MFM 8"
DW↑ bis DW↓	t_{WWCY}	µs	–	4	–	FM 5¼"
DW↑ bis DW↓			–	2	–	MFM 5¼"
DRQ↑ bis $\overline{RD}$↑/$\overline{WR}$↑	t_{RQRW}	µs	–	–	12	x[1]
DRQ↑ bis $\overline{RD}$↓	t_{RQR}	ns	800	–	–	x[1]
DRQ↑ bis $\overline{WR}$↓	t_{RQW}	ns	250	–	–	x[1]

x[1] doppelte Werte für U 8272 D04 (4 MHz)
x[2] doppelter Maximalwert für U 8272 D04

Tafel 10 Elektrische Kennwerte

Kenngröße	Symbol	Einheit	min.	typ	max.	Bedingung
L-Ausgangsspannung	U_{OL}	V	–	–	0,45	$I_{OL}=2\,mA$
H-Ausgangsspannung	U_{OH}	V	2,4	–	–	$I_{OH}=-200\,\mu A$
Eingangsleckstrom	$\|I_{LI}\|$	µA	–	–	10	$U_I=0{,}4\,V\ldots U_{CC}$
Ausgangsleckstrom	$\|I_{LO}\|$	µA	–	–	10	$U_O=0{,}45\,V\ldots U_{CC}$ Pin 6…13
$\overline{RD}$↓ bis Daten	t_{RD}	ns	–	–	200	
$\overline{RD}$↑ bis Daten hochohmig	t_{DF}	ns	20	–	100	
WRC↑ bis PSØ, PS1	t_{CP}	ns	20	–	100	
WRC↑ bis WRD↑	t_{CD}	ns	20	–	100	
WRC↑ bis WE↑	t_{WE}	ns	20	–	100	
WRD↑ bis WRD↓	t_{WDD}	ns	$t_{WCH}-50$	–	–	
Stromaufnahme	I_{CC}	mA	–	–	250	
Eingangskapazität	C_I	pF	–	–	10	
Eingangskapazität C, WRC	$C_I(C)$	pF	–	–	20	
Ein-Ausgangskapazität	$C_{I/O}$	pF	–	–	20	
$\overline{RD}$↑ bis INT↓	t_{RI}	ns	–	–	500	x[1]
$\overline{WR}$↑ bis INT↓	t_{WI}	ns	–	–	500	x[1]
DRQ↑ bis DRQ↑	t_{RQCY}	t_{CY}	104	–	–	
DAK↓ bis DRQ↓	t_{AKRQ}	ns	–	–	200	
FR↑ bis FR↓	t_{FR}	t_{CY}	64	–	80	
DSØ, DS1 bis $\overline{RW}$/SEEK↑	t_{US}	t_{CY}	96	–	–	
$\overline{RW}$/S↑ bis LC/DR	t_{SD}	t_{CY}	56	–	–	
DR bis ST↑	t_{DST}	t_{CY}	8	–	–	
ST↓ bis DR	t_{STD}	t_{CY}	192	–	–	
ST↓ bis DSØ, DS1	t_{STU}	t_{CY}	40	–	–	
ST↓ bis ST↓	t_{STP}	t_{CY}	–	40	–	
ST↑ bis ST↑	t_{SC}	t_{CY}	264	–	–	x[2]
$\overline{RW}$/S↓ bis DS0, DS1	t_{SU}	t_{CY}	120	–	–	
LC/DR bis $\overline{RW}$/S↓	t_{DS}	t_{CY}	240	–	–	

x[1] Diese Zeiten gelten nur für eine Taktfrequenz von 8 MHz. Bei 4 MHz verdoppeln sie sich.
x[2] Diese Zeit gilt für unterschiedliche Stationen. Für eine Station gilt der im Befehl SPECIFY angegebene Wert.
Als Bezugspegel für die Zeiten gelten 0,8 V (low) und 2,0 V (high).
Die Ausgänge sind mit 100 pF (einschließlich Meßkapazität) belastet. Die Einhaltung aller Kenngrößen wird vom Hersteller mit einer Toleranz von ±2,5 % garantiert.

Literatur

/1/ Köhler, Th., Streubel, R.: Diskettenspeicher – Stand und Tendenzen. Radio, Ferns., Elektron. Berlin 36 (1987) 1, S. 46–48
/2/ Alles über Disketten. Chip (1981) 4, S. 70–73
/3/ Böhl, E.: Die Floppy-Disk-Controller-Schaltkreise U 8272 D08 und U 8272 D04. Radio, Ferns., Elektron., Berlin 36 (1987) 11
/4/ Warninck, H.: Funktionsbeschreibung des Floppy-Disk-Controller-Schaltkreises U 8272. VEB Robotron Elektronik Dresden, 10/1986
/5/ Werkstandard, EMS 2633, VEB Forschungszentrum Mikroelektronik Dresden, November 1986

✉ KONTAKT ☎

VEB Forschungszentrum Mikroelektronik Dresden, Abt. EAP, Grenzstraße 28, Dresden, 8080; Tel. 588 246

Wegbereiter der Informatik

Friedrich Wilhelm Bessel

∗ 1784 Minden (Weser), † 1846 Königsberg (Kaliningrad).



Bessel war nicht nur einer der bedeutendsten Astronomen (er gilt als Pionier der astronomischen Meßkunst), sondern er hat auch auf mathematischem Gebiet herausragende Ergebnisse hinterlassen, die noch heute zum Lehrstoff in einem Mathematikstudium gehören. Bereits ein Blick in ein mathematisches Lexikon weist seine bleibenden Verdienste aus. Begriffe wie Besselsche Differentialgleichung, Bessel-Funktion, Besselsche Integrationsformel, Besselsche Polynome u. a. belegen diesen Sachverhalt. So hat er die Lösbarkeit einer nach ihm benannten Klasse von Differentialgleichungen untersucht, welche in der mathematischen Physik eine wichtige Rolle spielen. Er gab die Lösung in Form einer Potenzreihe an (Bessel-Funktionen), deren numerische Auswertung jedoch erst durch computertechnische Unterstützung mit vertretbarem Zeitaufwand hinreichend genau gelingt. Auch die von ihm abgeleiteten Formeln für die numerische Integration oder für die Interpolation einer Funktion haben praktisch den Charakter von Algorithmen für den unmittelbaren Einsatz von Computern.

Es ist bemerkenswert, daß Bessel während seiner Ausbildungszeit (er besuchte nach dem Gymnasium ab 1798 eine Bremer Navigationsschule) Mathematik lediglich autodidaktisch betrieb, allerdings so intensiv, daß er in deren praktischer Anwendung ungewöhnliche Fertigkeiten erlangte. Seiner ersten Publikation aus dem Jahre 1804 über den Halleyschen Kometen von 1607 liegen rund 300 Manuskriptseiten Berechnungen zugrunde. Diese Arbeit beeindruckte C. F. Gauß derart, daß er Bessel um Unterstützung bei der Realisierung der „Berechnung des geozentrischen Laufs 3 neuer Planeten" bat.

Wilhelm von Humboldt veranlaßten die Leistungen Bessels, im Jahre 1810 dessen Berufung zum Professor für Astronomie an die Königsberger Universität und zum Direktor der dort zu errichtenden Sternwarte zu betreiben. Bessel war übrigens der erste, der eine kosmische Entfernung jenseits der Grenzen des Sonnensystems exakt gemessen bzw. berechnet hat.

Dr. Klaus Biener

Moderne Mikrorechnersysteme (Teil 2)

Prof. Dr. Peter Neubert, Ralph Willem, Karsten Künne
Technische Universität Dresden, Sektion Informationstechnik

3. System 80186

Der Mikroprozessor 80186 stellt eine Weiterentwicklung des 8086 dar. Er erschien etwa im Jahr 1982. Intel entwickelte den 80186 vor allem, um das Kostenproblem, welches bei 16-Bit-Rechnern auftritt, zu lösen. Aus diesem Grunde wurden im 80186 alle wesentlichen Funktionen der ZVE eines Mikrorechners vereinigt. Man kann den 80186 fast schon als Einchipmikrorechner bezeichnen, nur der Speicher fehlt noch auf dem Chip.

Was enthält der 80186 nun im einzelnen:
- eine verbesserte 8 MHz-8086/2-CPU
- einen Taktgenerator
- zwei unabhängige DMA-Kanäle
- einen programmierbaren Interruptcontroller
- drei programmierbare 16-Bit-Zähler/Zeitgeber
- programmierbare Speicherselect- und Peripherieselectlogik
- einen programmierbaren Wartezustandsgenerator
- einen lokalen Buscontroller

Durch diese Vereinigung einer Vielzahl von Funktionen in einem Chip werden Gewicht, Größe, Leistungsaufnahme und Kosten bei steigender Leistung günstig beeinflußt. In Tafel 3 ist zur Untermauerung dieser Aussage das Leistungs-Kosten-Verhältnis verschiedener Intel-Prozessoren dargestellt.

Das System 80186 ist vollständig aufwärtskompatibel für Software und Peripherie des 8086. Obwohl der 80186 nicht so starke Verbreitung wie der 8086 erreicht hat, besitzt er auf dem OEM-Markt doch eine große Bedeutung.

Tafel 3 Leistung-Kosten-Verhältnis von Intel-MP /6/

	8085	8086	80186	80286
Leistung in MIPS	0,07	0,3	0,7	1,5
Leistung-Kosten-Verh. in IPS/$	700	1500	10000	7500

MIPS = 1 Million Instruktionen pro Sekunde

3.1. Bestandteile einer funktionsfähigen ZVE

Bedingt durch die große Anzahl zusätzlicher Funktionseinheiten im 80186 läßt sich eine funktionsfähige ZVE mit diesem Mikroprozessor bereits mit wenigen Schaltkreisen aufbauen. Bild 15 zeigt die einfachste Variante.

Außer dem Prozessor 80186 werden nur noch zwei Adreßlatches benötigt, da der 80186 die Adressen und Daten wie der 8086 im Multiplexbetrieb ausgibt. Soll die ZVE allerdings durch Koprozessoren, wie den Arithmetikprozessor 8087 oder den E/A-Prozessor 8089 erweitert werden, ist noch ein Buscontroller 82188 in das System zu integrieren, da der 80186 die Busübergabe nicht mit dem RQ/GT-Protokoll wie der 8086 im Maximummode abwickelt, sondern das HOLD/HLDA-Protokoll verwendet.

Die einfachste Variante nach Bild 1 bietet durch die integrierten Funktionen im 80186 bereits eine Reihe von Leistungsmerkmalen, die nicht unbedingt zu den Grundfunktionen einer ZVE gehören. An dieser Stelle sei darum kurz auf die zusätzlichen Funktionseinheiten des 80186 eingegangen.

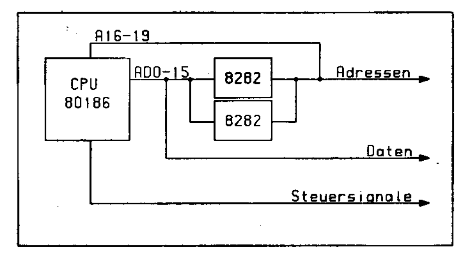


Bild 15 Einfache ZVE-Variante mit 80186 /6/

Zwei speziell für Echtzeitanwendungen wichtige Funktionseinheiten im 80186 sind der Interruptcontroller und der programmierbare Zähler/Zeitgeber. Der Interruptcontroller kann bis zu acht verschiedene Interruptquellen behandeln. Dabei werden simultane Interruptforderungen nach vorgegebenen Prioritäten aufgelöst. Es wird mit Vektorinterrupt gearbeitet, und eine selektive Sperrung von Interrupts ist möglich. Die Interruptquellen können sowohl extern (maximal fünf) als auch intern (Zähler/Zeitgeber oder DMA-Kanäle) sein. Eine Kaskadierung des Interruptcontrollers durch externe Controller 8259A ist möglich. Der programmierbare Zähler/Zeitgeber enthält drei 16-Bit-Kanäle, die für verschiedenste Funktionen programmiert werden können. Zwei der Timer können für externe Schaltungen verwendet werden, der dritte steht nur für interne Operationen zur Verfügung.

Die Funktionen des Zähler/Zeitgebers entsprechen etwa denen des Timerschaltkreises 8253.

Zur Unterstützung von schnellen Datentransfers enthält der 80186 zwei DMA-Kanäle. Die DMA-Kanäle ermöglichen Datenübertragungen mit einer Geschwindigkeit von maximal 2 MByte/s. Durch die interne Busvergabe sind die Reaktionszeiten auf eine DMA-Anforderung im Vergleich zu externen DMA-Controllern sehr kurz. Dagegen besitzen externe DMA-Controller gewöhnlich umfangreichere Programmiermöglichkeiten als die beiden internen DMA-Kanäle des 80186. Mit diesen ist es nur möglich, Datenblöcke zu übertragen.

Als weitere interne Funktionseinheit des 80186 sei noch auf die programmierbare Chipselect- und Ready-Logik eingegangen. Genau wie beim 8086 wird auch beim 80186 jeder Buszyklus durch ein Ready-Signal abgeschlossen. Dieses Ready-Signal kann zum einen extern erzeugt werden, aber es ist auch möglich, den internen Ready-Generator zu nutzen. Dieser Ready-Generator kann so programmiert werden, daß er zwischen null und maximal drei Wartezustände in den Buszyklus einfügt. Damit kann die Länge der Buszyklen optimal den Zugriffszeiten der Speicher bzw. E/A-Bausteine angepaßt werden. Weiterhin bietet der 80186 noch die Möglichkeit, Freigabesignale für Speicher und Peripherie durch die Chipselect-Logik zu bilden. Für die Selektierung des Speichers stehen insgesamt sechs Leitungen zur Verfügung. Die Speicherblockgröße, die jede Leitung adressiert, ist programmierbar und kann zwischen 1 KByte und 256 KByte liegen. Zur Selektierung von Peripheriebausteinen sind sieben Selectleitungen vorhanden. Jede Leitung ist dabei für ein 128-Byte-Segment im E/A-Bereich des Prozessors zuständig. Der Adreßbereich, in dem die Segmente liegen, ist wiederum frei programmierbar.

3.2. Unterschiede der CPU 80186 zur CPU 8086

Neben der Integration zusätzlicher Funktionseinheiten wurde beim 80186 auch die CPU gegenüber der CPU 8086 erheblich verbessert. Diese Verbesserungen betreffen vor allem die interne Struktur und den Befehlssatz.

Die Veränderungen in der internen Struktur beim 80186 dienen der Erhöhung der Verarbeitungsgeschwindigkeit. So wurde z. B. die Busanschlußeinheit mit einem eigenen Hardwareadder ausgestattet. Dadurch ist es möglich, die effektiven Adressen parallel zur Befehlsausführung zu berechnen. Der Multiplikationsteil wurde ebenfalls durch zusätzliche Hardware vergrößert und soll fünf- bis sechsmal schneller sein als der des 8086 /6/. Weiterhin wurde zur Beschleunigung von String-, Verschiebe- und Rotationsoperationen zusätzliche Hardware implementiert. Verschiebe- und Rotationsoperationen erfolgen dadurch mit einem Bit pro Takt.

Der Befehlssatz des 80186 wurde gegenüber dem 8086 um einige Befehle erweitert, die besonders der Unterstützung höherer Programmiersprachen dienen. Zu den neuen Befehlen gehören z. B. Befehle für Prozedurein- und -austritt, ein Befehl zum Testen von Indexgrenzen (erzeugt Interrupt 5 bei einem Fehler) und Block-E/A-Befehle. Weiterhin können beim 80186 Multiplikations-, Verschiebe- und Rotationsbefehle auch einen Direktoperanden verarbeiten, und es ist durch die Befehle PUSHA und POPA möglich, den Inhalt aller allgemeinen Register mit einem Befehl auf den Stack zu legen bzw. von dort zu holen. Bei Auftreten eines illegalen Operationscodes erzeugt der 80186 eine Befehlsausnahme (Interrupt 6), wodurch es möglich ist, zusätzliche Befehle zu generieren.

4. System 80286

Das Mikrorechnersystem 80286 stellt eine Weiterentwicklung des bekannten 8086-Systems dar. Es wurde dabei vor allem eine Erhöhung des Befehlsdurchsatzes, die Unterstützung von virtuellen Adressierungskonzepten, Verbesserung des Zugriffsschutzes und der Multitaskingfähigkeiten angestrebt. Bei gleicher Taktfrequenz soll der 80286 etwa 250 % der Leistung des 8086 erreichen /7/. Softwareseitig ist der 80286 weitgehend aufwärtskompatibel zum 8086. Das heißt, Programme, die für den 8086 geschrieben wurden, laufen problemlos auch auf dem 80286, nur mit entsprechend verkürzten Abarbeitungszeiten. Der maximale physische Adreßbereich beträgt 16 MByte, virtuell ist sogar 1 GByte verfügbar.

Der Prozessor 80286 enthält etwa 130000 Transistoren auf dem Chip, wird in HMOS-II-Technologie produziert und besitzt ein 68poliges Gehäuse (LCC oder PGA). Die Bedeutung des 80286 geht schon allein daraus hervor, daß mit ihm die IBM-PCs AT und XT286 ausgestattet sind und auch einige Modelle des neuen IBM-PS/2 auf diesem Prozessor basieren.

4.1. Bestandteile einer funktionsfähigen ZVE

Um eine funktionsfähige ZVE mit dem Mikroprozessor 80286 aufzubauen, benötigt man neben der CPU 80286 noch einen Taktgenerator 82284, einen Buscontroller 82288 und drei Adreßlatches 8282. Wenn noch ein leistungsfähiges Interruptsystem hinzugefügt werden soll, bietet sich die Verwendung des Interruptcontrollers 8259A an. Bei größeren Systemen werden außerdem noch zwei Treiber 8286 für den Datenbus notwendig. Im Bild 16 ist das Blockschaltbild einer funktionsfähigen ZVE mit dem 80286 gezeigt.
Der Taktgenerator liefert neben dem Systemtakt, der immer doppelt so groß wie der CPU-Takt ist, auch noch eine Synchronisation des Ready- und des Reset-Signals.

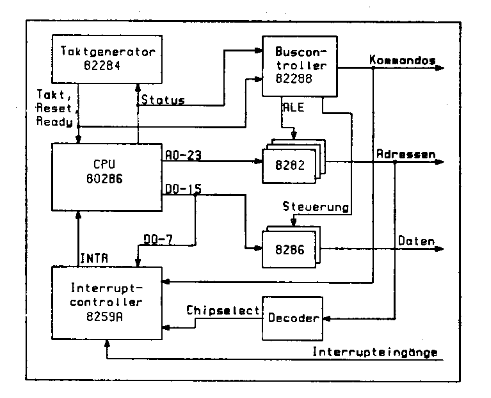
4.1.1. Multibusinterface

Wenn zu der im vorigen Abschnitt gezeigten funktionsfähigen ZVE noch ein Busverwalter 82289 hinzugefügt wird, entsteht ein vollständiges Multibusinterface. Der Busverwalter 82289 übernimmt dabei alle Aufgaben, die mit der Busvergabe in Zusammenhang stehen. Ein Blockschaltbild des 82289 ist in Bild 17 dargestellt.
Die Funktion des 82289 entspricht im wesentlichen der des 8289 aus dem 8086-System. Er fordert also bei Bedarf den Bus an und gibt den Controller und die Adreßlatches frei, wenn er die Busherrschaft erlangt hat. Benötigt der Prozessor den Bus nicht mehr, dann wird der Busverwalter durch einen anderen Busverwalter aus der Busherrschaft verdrängt usw. Die Prioritäten der Busverwalter am Multibus können parallel oder seriell (Daisy-Chain) gesteuert werden. Das ist identisch zum 8289. Um seine Aufgaben erfüllen zu können, verfügt der Busverwalter 82289 über Anschlüsse für alle Verwaltungsleitungen des Multibus und läßt sich auf der anderen Seite problemlos an den Prozessor 80286 anschließen. Bild 18 zeigt noch einmal eine ZVE mit vollständigem Multibusinterface.

4.1.2. Betriebsarten des Bussteuer-ICs 82288

Der Bussteuer-IC (Buscontroller) 82288 hat die Aufgabe, die Statussignale der CPU 80286 zu dekodieren und daraus die entsprechenden Lese- bzw. Schreibkommandos auf dem Bus zu erzeugen. Dazu erhält er von der CPU die beiden Statussignale S0 und S1 und das Signal M/IO. Die Kodierung des CPU-Status ist in Tafel 4 wiedergegeben.
Weiterhin steuert der 82288 die Adreßlatches und die Datenbustreiber durch entsprechende Steuersignale. Seine Funktion entspricht der des 8288 aus dem 8086-System. Bild 19 zeigt ein Blockschaltbild des 82288.
An dieser Stelle sei auf eine Besonderheit der Adreßausgabe des 80286 hingewiesen. Der 80286 verfügt über getrennte Busse für Daten und Adressen. Die Adressen jedes Buszyklus werden aber bereits vorzeitig ausgegeben, noch bevor der gerade laufende Buszyklus abgeschlossen ist. Das Taktdiagramm eines 80286-Buszyklus mit diesem Adreßpipelining ist in Bild 20 dargestellt.
Eine solche Überlappung von Buszyklen hat den Vorteil, daß mit bestimmten Schaltungsmaßnahmen den Speichern bzw. E/A-Bausteinen größere Antwortzeiten zur Verfügung gestellt werden können. Allerdings müssen die Adressen dann durch Latches gespeichert werden, da sie vor Beendigung des Buszyklus verschwinden.
Um nun das verbesserte Zeitverhalten der CPU 80286 stärker ausnutzen zu können, hat der Buscontroller 82288 zwei Betriebsarten, die sich im Zeitverhalten unterscheiden.
Die erste Betriebsart ist der Multibusmode. In dieser Betriebsart ist das Zeitverhalten des Buscontrollers multibuskompatibel. Auch die Steuerung der Datenbustreiber erfolgt so, daß sie dem Multibusstandard entspricht. Ein Multibuszyklus in dieser Betriebsart ist mindestens drei Prozessortakte lang, d. h., jeder Prozessorbuszyklus muß um mindestens einen Wartetakt (Waitstate) verlängert werden. Es wird also nicht die maximal mögliche Busbandbreite des Prozessors erreicht.
Eine zweite Betriebsart des Buscontrollers 82288 ist der lokale Busmode. Dabei werden die Schreib- bzw. Lesekommandos bereits einen ganzen bzw. einen halben Prozessortakt eher ausgegeben als im Multibusmode. Auch die Freigabe der Datenbustreiber erfolgt wesentlich zeitiger. In dieser Betriebsart kann die maximale Busbandbreite des Prozessors erreicht werden. Ein Buszyklus ist minimal zwei Prozessortakte lang, wenn kein Wartetakt eingefügt wird.
Die CPU 80286 kann auch mit zwei Bussystemen arbeiten. Dann sind allerdings auch zwei Buscontroller erforderlich und der physische Adreßraum muß zwischen beiden Bussystemen aufgeteilt werden. Ein Bus könnte beispielsweise der Multibus sein und der zweite ein schneller lokaler oder residenter Bus.

Bild 16 Einfache Variante einer ZVE mit dem 80286



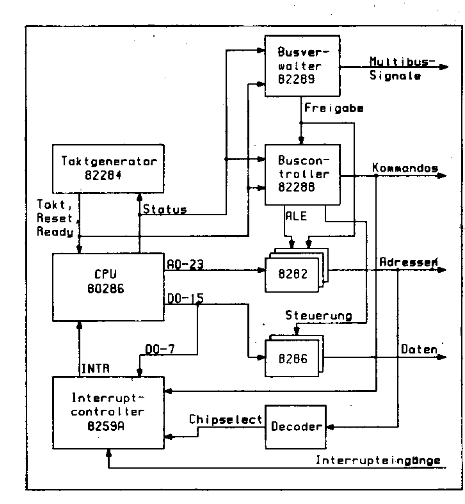


Bild 18 ZVE mit dem 80286 und einem Multibusinterface

Tafel 4 Statuskodierung des 80286 /8/

M/IO	*S1*	*S0*	*Typ des Buszyklus*
0	*0*	*0*	*Interruptquittung*
0	*0*	*1*	*E/A-Lesezyklus*
0	*1*	*0*	*E/A-Schreibzyklus*
0	*1*	*1*	*Leerzyklus*
1	*0*	*0*	*Halt oder Abschaltung*
1	*0*	*1*	*Speicherlesezyklus*
1	*1*	*0*	*Speicherschreibzyklus*
1	*1*	*1*	*Leerzyklus*

4.1.3. Interruptsystem

Das Interruptsystem des 80286 unterscheidet sich nicht wesentlich von dem des 8086. Die CPU 80286 verfügt über zwei Interrupteingänge, einen maskierbaren INTR und einen nichtmaskierbaren NMI. Der maskierbare Eingang kann durch das Interruptflag im Flagregister gesperrt oder freigegeben werden. Wenn dieser Eingang freigegeben ist und aktiviert wird, führt der Prozessor zwei Interruptbestätigungszyklen durch. Während des zweiten Zyklus liest er auf dem niederwertigen Teil des Datenbusses einen 8-Bit-Vektor ein, durch den dann die Startadresse einer Unterbrechungsbehandlungsroutine in einer Tabelle ausgewählt wird. Gewöhnlich wird der Interruptvektor durch einen Interruptcontroller 8259A geliefert, der bis zu acht Interruptquellen parallel behandeln kann. Durch Kaskadierung können bis maximal 64 Interruptquellen bearbeitet werden. Der Interruptcontroller 8259A arbeitet im 80286-System genauso wie im System 8086.
Bei Aktivierung des nichtmaskierbaren Interrupteingangs NMI werden keine Bestätigungszyklen durchgeführt, und es wird kein Vektor eingelesen. Dadurch ist die Reaktionszeit sehr kurz. Es kann allerdings auch nur eine Behandlungsroutine für diesen Interrupt geben.

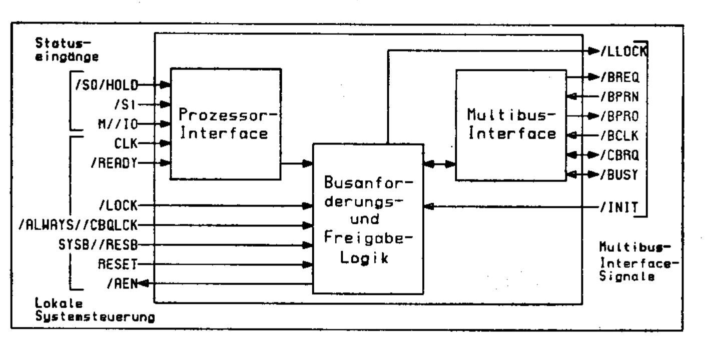


Bild 17 Blockschaltbild Busverwalter 82289 /3/

Fortsetzung siehe S. 211

Programmieren mit MACRO-SM

Teil V
Dr. Thomas Horn
Informatikzentrum des Hochschulwesens an der Technischen Universität Dresden

5.4. Datenformate

Adressen

Adressen werden wie vorzeichenlose ganze Zahlen (natürliche Zahlen) im Wortformat (16 Bit) gespeichert und verarbeitet. Durch die 16-Bit-Adressen ist der virtuelle Adreßraum auf 64 KByte beschränkt.

Intergerzahlen

Integerzahlen sind ganze Zahlen (mit Vorzeichen) im
- Byteformat (B) – 8 Bit
- Wortformat (I) – 16 Bit
- Langwortformat (L) – 32 Bit.

Grundsätzlich stehen alle arithmetischen und logischen Operationen für Integer-Zahlen im Wortformat zur Verfügung. Mit Ausnahme der Befehle für die arithmetischen Grundoperationen stehen auch alle Befehle für das Byteformat zur Verfügung. Für das Langwortformat ist nur ein Verschiebebefehl realisiert. Im Befehlssatz des FPP sind aber Konvertierungsbefehle zur Umwandlung in das D-Format vorhanden, so daß das L-Format effektiv über den FPP verarbeitet werden kann.
Ein Überlauf bei der Verarbeitung von Integerzahlen wird durch das V-Bit im Prozessorstatuswort (**PS**) angezeigt.

Natürliche Zahlen

Natürliche Zahlen sind ganze Zahlen ohne Vorzeichen im Byte- oder Wortformat. Für die Verarbeitung stehen die gleichen arithmetischen und logischen Befehle wie bei den Integerzahlen zur Verfügung. Ein auftretender Übertrag wird im C-Bit des **PS** angezeigt.

Gleitkommazahlen

Gleitkommazahlen werden durch FPP im
- F-Format (32 Bit) und
- D-Format (64 Bit)

verarbeitet. Es stehen umfangreiche arithmetische und Konvertierungsbefehle zur Verfügung. Bei Anlagen mit FIS sind nur 4 Befehle für das F-Format realisiert.
Eine Übersicht über die internen Darstellungen der Datenformate des SKR ist auf Seite 112 (MP 4/88) abgebildet.

5.5. Der Basisbefehlssatz des SKR

Der Basisbefehlssatz des SKR stellt den Standardbefehlsvorrat dar. Bei der Erläuterung der Befehle werden folgende Abkürzungen benutzt:

Mnemonischer Befehl

D – Ausdruck für die Adresse des Zieloperanden
S – Ausdruck für die Adresse des Quelloperanden
R – Registerausdruck
N – numerischer Ausdruck

numerischer Operationskode

DD – Zieloperandenfeld (6 Bit)
SS – Quelloperandenfeld (6 Bit)
R – Registerangabe (3 Bit)
XXX – Verschiebung (8 Bit), –128 bis +127
N – numerischer Wert (3 Bit)
NN – numerischer Wert (6 Bit)
NNN – numerischer Wert (8 Bit)

Operationsbeschreibung

d – Zieloperand
s – Quelloperand
r – Registerinhalt
(...) – Inhalt von ...
:= – Ergibtzeichen
\+ – Addition
– – Subtraktion, Vorzeichen (Zweierkomplement)
* – Multiplikation
/ – Division
~ – logische Negation (Einerkomplement)
∧ – logisches UND
∨ – inklusives logisches ODER
∀ – exklusives logisches ODER

Zu jedem Befehl wird die Aktualisierung der Flagbits des Prozessorstatusregisters (**PS**) angegeben, wobei folgende Symbolik verwendet wird:

* – wird entsprechend dem Ergebnis gesetzt oder gelöscht
– – wird nicht beeinflußt
0 – wird gelöscht
1 – wird gesetzt.

Das Setzen bzw. Löschen der Flagbits erfolgt, falls nicht anders angegeben, nach folgenden Grundregeln:

1. Das N- Bit (negtiv) wird dem Vorzeichenbit des Resultates entsprechend gesetzt (höchstwertiges Bit).
2. Das Z-Bit (zero) wird gesetzt, wenn alle Bits des Ergebnisses Null sind, anderenfalls wird es gelöscht.
3. Das V-Bit (overflow) wird bei einem arithmetischen Überlauf gesetzt (Verletzung des Zahlenbereiches der ganzen Zahlen).
4. Das C-Bit (carry) wird bei einem Übertrag gesetzt (Verletzung des Zahlenbereiches der natürlichen Zahlen).

Wenn zu einem Wortverarbeitungsbefehl ein äquivalenter Byteverarbeitungsbefehl realisiert ist, so erfolgt unter Angabe der Mnemonik beider Befehle die Abhandlung gemeinsam. Ein Byteverarbeitungsbefehl ist im mnemonischen Operationskode durch ein nachgestelltes „**B**" gekennzeichnet.

5.5.1. Allgemeine Befehle

CLR *D* **0050*DD*** *d*:=0 **Löschen** **N Z V C**
CLRB *D* **1050*DD*** (Clear) **0 1 0 0**
Der Zieloperand wird gelöscht.

COM *D* **0051*DD*** *d*:=~*d* **Komplement** ****01**
COMB *D* **1051*DD*** (Complement)
Der Zieloperand wird logisch komplementiert (Einerkomplement), d. h., jede Binärziffer wird negiert. Das C-Bit wird gesetzt!

INC *D* **0052*DD*** *d*:=*d*+1 **Inkrement** *****–**
INCB *D* **1052*DD*** (Increment)
Zum Zieloperanden wird eine 1 addiert. Das C-Bit wird nicht verändert!

DEC *D* **0054*DD*** *d*:=*d*–1 **Dekrement** ********
DECB *D* **1054*DD*** (Decrement)
Vom Zieloperanden wird eine 1 subtrahiert. Das Bit wird nicht verändert!

NEG *D* **0053*DD*** *d*:=–*d* **Negation** ********
NEGB *D* **1053*DD*** (Negate)
Vom Zieloperanden wird das Zweierkomplement gebildet (Subtraktion von 0). Das V-Bit wird gesetzt, wenn die größte negative Zahl negiert wird, da sie kein positives Äquivalent hat und die Operation nicht ausgeführt werden kann. Das C-Bit wird prinzipiell mit einer Ausnahme gesetzt, wenn der Wert Null negiert wird.

TST *D* **0057*DD*** *d*–0 **Test** ****00**
TSTB *D* **1057*DD*** (Test)
Der Zieloperand wird getestet, wobei **N**- und Z-Bit entsprechend gesetzt oder gelöscht werden.

SWAB *D* **0003*DD*** **Vertauschen der Bytes** (Swap bytes) ****00**
Höher- und niederwertiges Byte des Zieloperanden werden miteinander vertauscht. Die Zieladresse muß eine Wortadresse sein. Das N-Bit wird gesetzt, wenn im Ergebnis das Vorzeichen des niederwertigen Bytes (Bit 7) Eins ist. Das Z-Bit wird gesetzt, wenn im Ergebnis das niederwertige Byte Null ist.

MOV *S,D* **01*SSDD*** *d*:=*s* **Transport** ****0–**
MOVB *S,D* **11*SSDD*** (Move)
Der Quelloperand wird auf die Zieladresse transportiert. Der Quelloperand und das C-Bit bleiben unverändert. Das N- und Z-Bit werden entsprechend dem transportierten Operanden gesetzt. Bei MOVB mit Registeradressierung für die Zieladresse ist zu beachten, daß das höherwertige Byte mit dem Vorzeichen des geladenen niederwertigen Bytes gefüllt wird (Sign extension).

CMP *S,D* **02*SSDD*** *s*–*d* Vergleich ********
CMPB *S,D* **12*SSDD*** (Compare)
Vergleich des Quelloperanden mit dem Zieloperanden durch Subtraktion des Zieloperanden vom Quelloperanden. Die Operanden bleiben unverändert. Das V-Bit wird bei einem arithmetischen Überlauf gesetzt (siehe **SUB**-Befehl). Das C-Bit wird bei einem Übertrag vom höchstwertigen Bit gesetzt. Beim Vergleich von natürlichen Zahlen ist das C-Bit gesetzt, wenn der Quelloperand kleiner als der Zieloperand ist.

5.5.2. Arithmetische Befehle (Festkomma)

ADD *S,D 06SSDD d*:=*d*+*s* **Addition (Add)** ********
Der Quelloperand wird zum Zieloperanden addiert. Der Zieloperand wird dabei durch das Resultat überschrieben, der Quelloperand bleibt unverändert. Das C-Bit wird bei einem Übertrag vom höchstwertigen Bit gesetzt. Das V-Bit wird beim arithmetischen



Überlauf gesetzt (wenn beide Operanden das gleiche Vorzeichen haben und das Ergebnis das inverse Vorzeichen hat).

SUB ***S,D*** **16*SSDD*** *d:=d−s* **Subtraktion** (Subtract) ****

Der Quelloperand wird vom Zieloperanden subtrahiert. Das C-Bit wird bei einem Übertrag vom höchstwertigen Bit gesetzt. Das V-Bit wird bei arithmetischem Überlauf gesetzt (Minuend und Subtrahend haben verschiedene Vorzeichen und das Vorzeichen des Resultats ist mit dem Vorzeichen des Subtrahenden identisch.

ADC ***D*** **0055*DD*** *d:=d+C* **Addition des C-Bits** ****
ADCB ***D*** **1055*DD*** (Add carry)

Das C-Bit wird zum Zieloperanden addiert.

SBC ***D*** **0056*DD*** *d:=d−C* Subtraktion des C-Bits ****
SBCB ***D*** **1056*DD*** (Subtract carry)

Das C-Bit wird vom Zieloperanden subtrahiert.

5.5.3. Logische Befehle

BIT ***S,D*** **03*SSDD*** *s∧d Bit Testen* **0−
BITB ***S,D*** **13*SSDD*** (Bit test)

Verknüpft die beiden Operanden konjunktiv und setzt entsprechend N- und Z-Bit. Die Operanden werden nicht verändert. Gewöhnlich werden durch den Befehl mittels einer Maske ein oder mehrere Bits des Zieloperanden getestet.

BIC ***S,D*** **04*SSDD*** *d(:~S)∧d* **Bit Löschen** **0−
BICB ***S,D*** **14*SSDD*** (Bit clear)

Es werden im Zieloperanden die Bits auf Null gesetzt, die im Quelloperanden (Maske) mit Eins belegt sind (Konjunktion mit negiertem Quelloperand).

BIS ***S,D*** **05*SSDD*** *d:=s∨d* **Bit Setzen** **0−
BISB ***S,D*** **15*SSDD*** (Bit set)

Es werden im Zieloperanden die Bits auf Eins gesetzt, die im Quelloperanden mit Eins belegt sind (Disjunktion).

5.5.4. Verschiebebefehle

ROR ***D*** **0060*DD*** **Zyklische Rechtsverschiebung** ****
RORB ***D*** **1060*DD*** (Rotate right)

Alle Bits des Zieloperanden werden um 1 Bit nach rechts verschoben. Das C-Bit wird in das höchstwertige Bit und Bit 0 in das C-Bit eingetragen (Bild 17).

Das V-Bit wird wie folgt gesetzt: V = N∀C. Damit zeigt das V-Bit an, daß der neue Wert des C-Bits gegenüber dem vorhergehenden geändert ist.

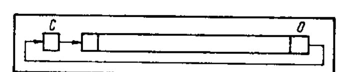


Bild 17 Zyklische Rechtsverschiebung

ROL ***D*** **0061*DD*** **Zyklische Linksverschiebung** ****
ROLB ***D*** **1061*DD*** (Rotate left)

Alle Bits des Zieloperanden werden um 1 Bit nach links verschoben. Das C-Bit wird in das Bit 0 und das höchstwertige Bit in das C-Bit eingetragen. Das V-Bit wird wie beim **ROR**-Befehl gesetzt (Bild 18).

ASR ***D*** **0062*DD*** **Arithmetische Rechtsverschiebung** ****
ASRB ***D*** **1062*DD*** **(Arithmetic shift right)**

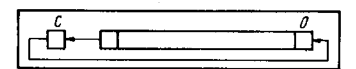


Bild 18 Zyklische Linksverschiebung

Alle Bits des Zieloperanden werden um 1 Bit nach rechts verschoben. Das höchstwertige Bit bleibt unverändert. Bit 0 wird in das C-bit geladen. Dieser Befehl entspricht einer Division durch 2. Das V-Bit wird wie beim ROR-Befehl gesetzt (Bild 19).

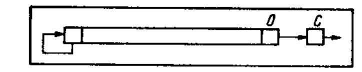


Bild 19 Arithmetische Rechtsverschiebung

Hinweis:
Bei einem Byteverarbeitungsbefehl mit einem Registeroperanden bleibt das höherwertige Byte unverändert (Ausnahme: **MOVB**).

ASL ***D*** **0063*DD*** **Arithmetische Linksverschiebung** ****
ASLB ***D*** **1063*DD*** **(Arithmetic shift left)**

Alle Bits des Zieloperanden werden um 1 Bit nach links verschoben. Das höchstwertige Bit wird in das C-Bit geladen. In das Bit 0 wird eine Null geladen. Solange das V-Bit nicht gesetzt wird, entspricht die Operation einer Multiplikation mit 2. Das V-Bit wird wie beim **ROR**-Befehl gesetzt und zeigt eine Vorzeichenänderung an (Bild 20).

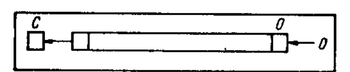


Bild 20 Arithmetische Linksverschiebung

5.5.5. Verzweigungsbefehle

Alle Verzweigungsbefehle sind 1-Wort-Befehle, die im niederwertigen Byte eine Wortverschiebung *XXX* (relative Adresse gemäß 5.3.3.) enthalten. Der Sprungbereich ist somit relativ zum aktualisierten **PC** (alter **PC** plus 2) −128 Worte rückwärts und +127 Worte vorwärts. Die Wortverschiebung wird im Befehlskode mit *XXX* dargestellt (*XXX* = *D* − **PC**).

Allgemeine Verzweigungsbefehle

BR	***D*** **000400+*XXX***	unbedingt (Branch)	
BNE	***D*** **001000+*XXX***	bei ungleich Null (Branch if not equal)	Z=0
BEQ	***D*** **001400+*XXX***	bei gleich Null (Branch if equal)	Z=1
BPL	***D*** **100000+*XXX***	bei Plus (Branch if plus)	N=0
BMI	***D*** **100400+*XXX***	bei Minus (Branch if minus)	N=1
BVC	***D*** **102000+*XXX***	bei kein Überlauf (Br. if overflow clear)	V=0
BVS	***D*** **102400+*XXX***	bei Überlauf (Br. if overflow is set)	V=1
BCC	***D*** **103000+*XXX***	bei kein Übertrag (Br. if carry is clear	C=0
BCS	***D*** **103400+*XXX***	bei Übertrag (Br. if carry is set)	C=1

Hinweis:
Die Anwendung von **BPL/BMI** nach dem **CMP/CMPB**-Befehl und anderen arithmetischen Befehlen führt bei Überschreitung des Wertebereiches zu Fehlern, da das C- bzw. V-Bit vom Verzweigungsbefehl nicht analysiert wird. In Abhängigkeit davon, ob die 16-Bit-Zahlen als ganze Zahlen oder natürliche Zahlen interpretiert werden, sollten die Verzweigungsbefehle der nachfolgenden Gruppen Anwendung finden.

Verzweigungsbefehle nach Operationen mit ganzen Zahlen

BGE	***D*** **002000+*XXX***	bei größer oder gleich Null (Branch if greater than or equal)	N∀V=0
BLT	***D*** **002400+*XXX***	bei kleiner als Null (Branch if less than)	N∀V=1
BGT	***D*** **003000+*XXX***	bei größer als Null (Branch if greater than)	Z∨(N∀V)=30
BLE	***D*** **003400+*XXX***	bei kleiner oder gleich Null (Branch if less than or equal	Z∨(n∀V)=1

Diese Befehle werden immer nach arithmetischen Operationen mit ganzen Zahlen angewendet, da sonst nach einer möglichen Überschreitung des Wertebereiches falsch verzweigt wird.

Verzweigungsbefehle nach Operationen mit natürlichen Zahlen

BHI	***D*** **101000+*XXX***	Bei größer (Branch if higher)	C∨Z=0
BLOS	***D*** **101400+*XXX***	bei kleiner oder gleich (Branch if lower or same)	C∨Z=1
BHIS	***D*** **103000+*XXX***	bei größer oder gleich (Branch if higher or same)	C=0
BLO	***D*** **103400+*XXX***	bei kleiner (Branch if lower)	C=1

Diese Befehle werden nach arithmetischen Operationen mit natürlichen Zahlen (Adressen) angewendet, da sonst nach einer möglichen Überschreitung des Wertebereichs falsch verzweigt wird.

5.5.6. Sprungbefehle und Befehle zur Unterprogrammorganisation

JMP ***D*** **0001*DD*** Unbedingter Sprung (Jump)

Der **JMP**-Befehl gestattet einen Sprung zu einer beliebigen Adresse, da außer der Registeradressierung alle Adressierungsmodifikationen zulässig sind. Die Zieladresse muß immer geradzahlig sein. Die Adressierungsmodikation 0 bewirkt einen Interrupt nach Adresse 000010 (Verbotener Befehl) und eine ungerade Adresse 000004 (Time out / Verbotene Adresse). Die Flags des PS werden nicht verändert.

Operation: **PC**:=*d*

JSR ***R,D*** **004*RDD*** **Unterprogrammsprung** (unbedingt) (Jump to subroutine)

Der **JSR**-Befehl gestattet einen Sprung zu einem Unterprogramm mit einer bestimmten Anfangsadresse. Vor dem Sprung wird der alte PC-Inhalt in das Register R gerettet, um eine Rückkehr zum Hauptprogramm zu ermöglichen. Damit der Inhalt des Registers R nicht verloren geht, wird er zuvor in den Systemstack gerettet. Der Sprung wird dann analog dem **JMP**-Befehl ausgeführt.

Operation:

tmp:=d	(Die Zieladresse wird berechnet und in ein internes temporäres Register gespeichert)
−(SP):=*R*	(Inhalt von R wird gerettet)
R:=**PC**	(**PC** wird nach R gerettet)
PC:=*tmp*	(Sprung wird ausgeführt)

RTS ***R*** **00020*R*** **Rücksprung aus dem Unterprogramm**

Vom Register *R* wird die Rücksprungadresse geladen. Der Inhalt des Registers *R* wird vom Stack geladen. Die Flags des **PS** werden nicht beeinflußt.

Operation: **PC**:=***R*** (Laden der Rücksprungadresse von *R)*
R:=**(SP)+** (Laden des alten Registerinhaltes vom Stack)

5.5.7. *Steuerbefehle*

HALT 000000 Halt

Der Prozessor geht in den statischen Haltzustand. Die Datenanzeige des Bedienpultes gibt den Inhalt des Registers **R0** an. Die Adreßanzeige gibt die Adresse des nächsten Befehls an. Durch Drücken der Fortsetzungstaste (продолжать/Continue) kann das Programm fortgesetzt werden. Im Nutzer-Modus ruft der Befehl eine Unterbrechung hervor.

WAIT 000001 Warten auf Interrupt (Wait for interrupt)

Der Prozessor geht in den dynamischen Stop und erwartet ein Interrupt. Der **PC** zeigt auf den nächsten Befehl, so daß nach der Interruptbehandlungsroutine (**RTI**-Befehl) der nächste Befehl nach **WAIT** ausgeführt werden kann. Im Nutzermodus wird der Befehl als **NOP**-Befehl ausgeführt.

RESET 000005 Rücksetzen des externen Busses (Reset external Bus)

Der Prozessor sendet auf den Einheitsbus für 10 ms das Signal INIT. Dadurch werden alle Geräte in ihren Grundzustand rückgesetzt. Im Nutzermodus wird der Befehl als **NOP**-Befehl ausgeführt.

RTI 000002 Rückkehr von der Interruptbehandlungsroutine (Return from Interrupt)

PC und **PS** werden vom Systemstack geladen. Es wird der Rücksprung in das unterbrochene Programm organisiert sowie der alte Flagzustand eingestellt.

Operationen:

PC:=(SP)+	(Laden der Rücksprungadresse aus dem Systemstack)
PS:=(SP)+	(Laden des alten PS aus dem Systemstack)

RTT 000006 Rückkehr von der Trapbehandlungsroutine (return from trap)

Der Befehl wird wie der RTI-Befehl ausgeführt. Der Unterschied ist, daß der **RTT**-Befehl den T-Bit-Trap blockiert, so daß es bei gesetztem T-Bit **im PS** erst nach dem nächsten Befehl wirksam wird. Beim **RTI**-Befehl würde der T-Bit-Trap sofort wirksam werden. Dieser Befehl ist ein Zusatzbefehl für Rechner mit Hauptspeicherverwaltungseinheit. Bei Rechenanlagen vom Typ SM3 wird der T-Bit-Trap auch beim **RTI**-Befehl erst nach einem Befehl wirksam.

NOP	**000240**	**Keine Operation** (No operation)
CLC	**000241**	Löschen des C-Bits (Clear C)
CLV	**000242**	Löschen des V-Bits (Clear V)
CLZ	**000244**	Löschen des Z-Bits (Clear Z)
CLN	**000250**	Löschen des N-Bits (Clear N)
CCC	**000257**	Löschen aller Flags (Clear all condition codes)
SEC	**000261**	Setzen des C-Bits (Set C)
SEV	**000262**	Setzen des V-Bits (Set V)
SEZ	**000264**	Setzen des Z-Bits (Set Z)
SEN	**000270**	Setzen des N-Bits (Set N)
SCC	**000277**	Setzen aller Flags (Set all condition codes)

Die letzten vier Bits der Operationskodes stimmen der Bedeutung nach mit den Flagbits des **PS** überein. Es ist deshalb möglich, numerische Operationskodes zum Löschen/Setzen von zwei bzw. drei Flags zu bilden, die vom Assembler nicht unterstützt werden, z. B.:

000256 Löschen vom **V**-, **Z**- und N-Bit

EMT N 104000+*NNN* **Emulatortrap-Befehl**

TRAP N 104400+*NNN* **Trap-Befehl**

Beide Befehle rufen eine synchrone Unterbrechung (Trap) hervor und werden gleich ausgeführt.

Operation:

−(SP):=PS	(Retten des alten **PS**)
−(SP):=PC	(Retten der Rücksprungadresse)
PC:=(*IV*)	(Laden der Adresse der Trap-Behandlungsroutine vom Interruptvektor ***IV***)
PS:=(*IV*+2)	(Laden des neuen PS vom Interruptvektor ***IV***+2)

Der Interruptvektor für den **EMT**-Befehl steht auf Adresse 30(8) und für **den TRAP**-Befehl auf Adresse 34(8). Der **EMT**-Befehl ist für die Systemsoftware reserviert und darf vom Anwenderprogrammierer nicht benutzt werden. Der **TRAP**-Befehl steht für die allgemeine Verwendung zur Verfügung.

BPT 000003 Trap-Befehl für Unterbrechungspunkte (Breakpoint trap)

Dem **EMT**- und **TRAP**-Befehl entsprechend ruft dieser Befehl eine synchrone Unterbrechung hervor. Der Interruptvektor für den **BPT**-Befehl steht auf der Adresse 14(8). Der T-Bit-Trap nutzt den gleichen Interruptvektor. Der **BPT**-Befehl und der T-Bit-Trap werden vom Testhilfesystem ODT des Betriebssystems genutzt.

IOT 000004 Trap-Befehl für die Ein- und Ausgabe (I/O-Trap)

Der **IOT**-Befehl wird dem **EMT**- und **TRAP**-Befehl entsprechend ausgeführt. Sein Interruptvektor steht auf der Adresse 20(8). Er wurde vom E/A-Ruf der Lochbandsoftware genutzt.

MFPI ***S*** **0065*SS*** **Transport vom vorhergehenden Befehlsadreßraum** **0– (Move from previous Instruction Space)

Der Befehl ist ein Zusatzbefehl bei Rechnern mit Hauptspeicherverwaltungseinheit. Er gestattet das Lesen von einzelnen Worten aus dem vorhergehenden Befehlsadreßraum. Das gelesene Wort wird in den Stack geschrieben.

Operation:
tmp :=*S*
−(SP) :=*tmp*

Die Befehls- und die Stackadresse werden entsprechend dem aktuellen Satz der Segmentadreßregister (Bit 14, 15 des **PS**) auf den physischen Speicher abgebildet, während für die Quelladresse der Segmentadreßregistersatz gemäß Bit 12, 13 des **PS** genutzt wird.

MTPI ***D*** **0066*DD*** **Transport in den vorhergehenden Befehlsadreßraum** **0– (Move to previous Instruction Space)

Der Befehl ist die Umkehrung des **MFPI**-Befehls. Er gestattet das Schreiben von einzelnen Worten in den vorhergehenden Befehlsadreßraum. Das zu schreibende Wort wird aus dem Stack gelesen:

Operation:
tmp := **(SP)+**
D :=*tmp*

Die Befehls- und die Stackadresse werden entsprechend dem aktuellen Satz der Segmentadreßregister (Bit 14, 15 des PS) auf den physischen Speicher abgebildet, während für die Zieladresse der Segmentadreßregistersatz gemäß Bit 12, 13 des **PS** genutzt wird.

SPL ***N*** **00023*N*** **Setzen der Priorität** (Set priority level) ––––

Der Befehl ist ein Zusatzbefehl beim K 1620 und bei der Elektronika-60. Die niederwertigen drei Bits des Befehls werden als Prozessorpriorität in das **PS** geladen. Im Nutzermodus wird der Befehl als **NOP**-Befehl ausgeführt.

MFPS ***D*** **1067*DD*** **Speichern des PS** (Move byte from Processor status word) **0–

Der Befehl ist ein Zusatzbefehl des K 1620 und der Elektronika-60. Die niederwertigen 8 Bits des PS werden unter der Zieladresse abgespeichert. Die Zieladresse wird als Byteadresse interprtiert. Das Z-Bit wird gesetzt, wenn alle Bits im niederwertigen Byte des **PS** null sind.

MTPS ***S*** **1064*SS*** **Laden des PS** (Move byte to processor status word) ****

Der Befehl ist ein Zusatzbefehl des K 1620 und der Elektronika-60. Die Bit 0–3 des Quelloperanden werden in das **PS** geladen. Die Quelloperandenadresse wird als Byte-

adresse interpretiert. Das T-Bit kann mit diesem Befehl nicht gesetzt werden.

5.6. Der erweiterte Befehlssatz des SKR

MUL *S,R* 070*RSS* (*R,R*V1):=*r*s*** ****0*** **Multiplikation** (Multiply)

Es werden zwei ganze Zahlen (16-Bit) multipliziert, wobei ein 32-Bit-Produkt entsteht. Wenn *R* geradzahlig ist, wird im geradzahligen Register der höherwertige Teil des Produkts und im darauffolgenden ungeradzahligen Register der niederwertige Teil gespeichert. Wenn *R* ungeradzahlig ist, so wird nur der niederwertige Teil des Produkts gespeichert. Das C-Bit wird gesetzt, wenn das Ergebnis mehr als 15stellig wird (Das C-Bit muß, vorher gelöscht werden!).

Beispiele:

```
MOV ALPHA,R0  ;LADEN DES ERSTEN FAKTORS
MUL M0,R0     ;MULTIPLIZIEREN MIT K0
              ;ERGEBNIS IN R0,R1
CLC           ;LOESCHEN DES C-BITS
MOV FELD(R5),
           R3 ;LADEN DES
              ;ERSTEN FAKTORS
MUL FELD+2
     (R5),R3  ;MULTIPLIZIEREN
BCS FEHLER    ;---→ UEBERLAUFFEHLER
```

Hinweis:
Bei den Zusatzbefehlen **MUL**, **DIV**, **ASH** und **ASHC** ist die Operandenfolge im Assemblerformat gegenüber dem numerischen Maschinenbefehl vertauscht.

DIV *S,R* 071*RSS* (*R,R*+1):=(*R,R*+1)/s ******** **Division** (Divide)

Es wird ein 32stelliger Dividend (Ganze Zahl) durch einen 16stelligen Divisor dividiert. Der Quotient (16 Bit) wird im Register *R* und der Divisionsrest im Register *R*+1 gespeichert. Der Rest hat das gleiche Vorzeichen wie der Dividend. Das Register *R* muß immer geradzahlig sein. Das V-Bit wird gesetzt, wenn der Divisor 0 ist oder der Quotient größer als 15 Bit wird. Das C-Bit wird gesetzt, wenn durch 0 dividiert werden soll, anderenfalls wird es gelöscht.

Beispiele:

```
CLR R2          ;BEI NATUERLICHEN
                ;ZAHLEN ZULÄSSIG
MOV #5000.,R3   ;LADEN DES DIVIDENDEN
DIV K2,R2       ;DIVIDIEREN DURCH K2
..
MOV
    GAMMA,R3    ;LADEN DES
                ;16-BIT-DIVIDENDEN
                ;(GANZZAHLIG)
SXT R2          ;LADEN R2 MIT
                ;VORZEICHEN VON R3
DIV 10.,R2      ;DIVIDIEREN DURCH 10.
```

SXT *D* 0067*DD* Vorzeichen ausdehnung (Sign extension) **-*0-**

Bei N=0 wird der Zieloperand gelöscht. Bei N=1 wird der Zieloperand auf −1 gesetzt. Dieser Befehl wird insbesondere bei zur Umwandlung einer ganzen Zahl aus dem Wortformat in das Doppelwortformat benutzt (Beispiel siehe **DIV**-Befehl).

XOR *R,D* 074*RDD* *d:=r*∀*d* **Exclusives ODER** (Exclusive OR) ****0-**

Der Registerinhalt und der Zieloperand werden nach der Funktion „Exklusiv-ODER" (Antivalenz) verknüpft und das Resultat unter der Zieladresse gespeichert.

Beispiel:

```
XOR     R2,SYM0
vorher:  (R2)=0123757 (SYM0)=0137744 1 0 1 1
nachher: (R2)=0123757 (SYM0)=0014013 0 0 0 1
```

ASH *S,R* 072*RSS* Arithmetische Verschiebung (Arithmetic shift) ********

Das Register *R* wird arithmetisch nach rechts oder links verschoben, wobei die Richtung und die Anzahl der Verschiebeschritte durch die 6 niederwertigen Bits des Quelloperanden festgelegt werden. Die artihmetische Verschiebung wird analog dem ASR/ASL-Befehl durchgeführt. Das C-Bit enthält immer das letzte herasusgeschobene Bit. Das V-Bit wird gesetzt, wenn während der Linksverschiebung ein Vorzeichenwechsel eintritt. Ein positiver Quelloperand bewirkt eine Linksverschiebung und ein negativer eine Rechtsverschiebung. Der Wertevorrat des Quelloperanden liegt im Bereich von −32. bis +31.

Beispiele:

```
ASH #4,R2      ;LINKSVERSCHIEBUNG
               ;UM 4 BIT
ASH SHIFT,R0   ;DIE VERSCHIEBUNG
               ;VON R0 WIRD DURCH DEN
               ;INHALT VON SHIFT DEFINIERT
```

ASHC *S,R* 073*RSS* Kombinierte arithmetische Verschiebung (Arithmetic shift combined) ********

Wenn *R* geradzahlig ist, wird das Doppelregister *R,R*+1 analog dem **ASH**-Befehl verschoben. Es erfolgt ein Übertrag zwischen Bit 15 von *R*+1 und Bit 0 von *R* und umgekehrt. (Weitere Hinweise siehe ASH-Befehl.) Wenn *R* ungeradzahlig ist, wird das ungeradzahlige Register entsprechend dem Wert des Quelloperanden zyklisch (über 16 Bit) verschoben; da ein direkter Übertrag zwischen Bit 0 und Bit 15 besteht. Das C-Bit enthält den letzten Übertrag zweischen beiden Bits. Das V-Bit wird bei Vorzeichenwechsel gesetzt.

Beispiel:

```
ASHC #-5,R4    ;(R4,R5) WERDEN UM 5 BIT
               ;NACH RECHTS VERSCHOBEN
               ;DIVISION DURCH 32
ASCH (R2)+,R1  ;R1 WIRD ZYKLISCH VER-
               ;SCHOBEN (R2)+ LEGT RICH-
               ;TUNG UND ANZAHL FEST
```

MARK *N* 0064*NN* Unterprogramm-Rücksprungunterstützung (Mark) **----**

Dieser Befehl wird durch die Standardunterprogramme benutzt. Er gestattet die Säuberung des Stacks und hilft bei der Organisation des Rücksprungs. Der Wert *NN* im Befehlskode gibt die Anzahl der im Stack zu löschenden Parameter an.

Operation:

SP:=PC+2**NN*** (Laden der Adresse des alten Inhaltes des Registers **R5)**
PC:=R5 (Laden der Rücksprungadresse aus **R5)**
R5:=(SP)+ (Laden des **R5** mit dem alten Inhalt)

N Z V C

Beispiele:

```
        MOV R5,-(SP)    ;DEN INHALT
                        ;VON R5 RETTEN
        MOV P1,-(SP)    ;LADEN VON N
                        ;PARAMETER FUER DAS
        MOV P2,-(SP)    ;UNTERPROGRAMM
                        ;IN DEN STACK
        ...
        MOV PN,-(SP)
        MOV AMRKN,-(SP) ;LADEN DES BEFEHLS
                        ;MARK N IN DEN STACK
        MOV SP,R5       ;LADEN DER ADRESSE
                        ;DES MARK-BEFEHLS
        ...
        JSR PC, UPN     ;AUSFUEHREN DES UP
AMRKN:  MARK N          ;DEFINITION DES
                        ;MARK-BEFEHLS
        ...
UPN:    ...             ;ANFANG DES UP
        ...
        RTS R5          ;RUECKSPRUNG INS
                        ;HAUPTPROGRAMM
```

Bei dem **RTS-R5**-Befehl erfolgt der Sprung in den Stack zur Ausführung des **MARK**-Befehls. Dabei wird in **R5** aus dem Stack die Rücksprungsadresse geladen. Beim **MARK**-Befehl erfolgen dann die Korrektur des **SP** auf die Adresse vor der Rücksprungadresse und das Wiederherstellen des alten Inhaltes von **R5**.

SOB *R,D* 077*RNN* Zyklusbefehl (Subtract one and branch) **N Z V C**

Mit dem **SOB**-Befehl lassen sich kleinere Schleifen organisieren. Vom Register *R* wird eine Eins subtrahiert. Wenn das Ergebnis ungleich Null ist, ist die Schleife noch nicht beendet und die Verschiebung wird doppelt vom **PC** subtrahiert (Wortverschiebung!). Es ist somit nur ein Rückwärtssprung um max. 63 Worte möglich.

Beispiel:

```
      MOV #20.,R1  ;20. DURCHLAEUFE
2○:   ...          ;BEGINN DER SCHLEIFE
      ...
      SOB R1,2○    ;TEST AUF SCHLEIFEN-
                   ;ENDE
```

Der vorliegende MP-Kurs versuchte eine Einführung in die Makroassemblersprache MACRO-SM für die Klein- und Mikrorechner des SKR zu geben. Im Abschnitt 5 wurde vor allem auf die Adressierungsarten sowie die Basisbefehle und den erweiterten Befehlssatz eingegangen, die von den meisten Mikroprozessorschaltkreisfamilien realisiert werden. Nicht eingegangen werden konnte auf die 46 Befehle des Gleitkommaprozessors, über den die meisten Kleinrechenanlagen verfügen. Hierzu sei auf die einschlägige Fachliteraturr (Teil I) und die Dokumentationen verwiesen. Auch mußte leider aus Platzgründen auf das eingangs angekündigte abschließende Programmierbeispiel verzichtet werden.

Schluß

Fortsetzung von S. 206

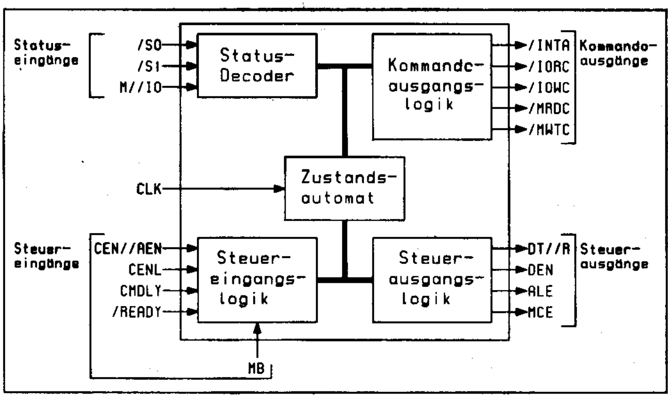


Bild 19 Blockschaltbild Buscontroller 82288 /12/

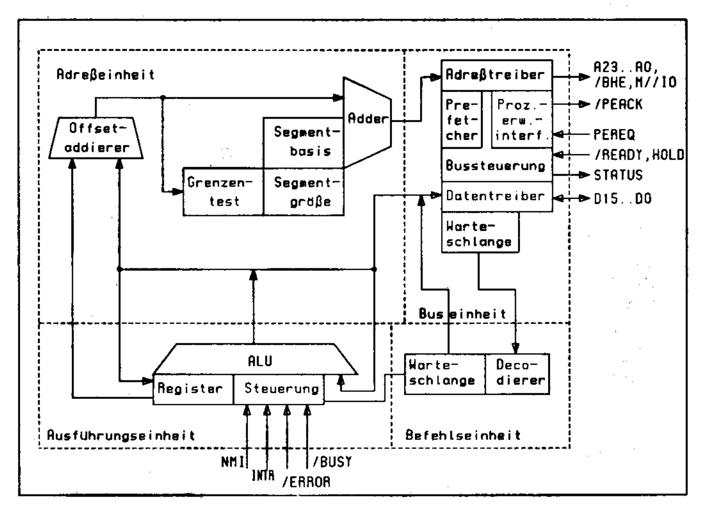


Bild 21 Blockschaltbild des 80286 /9/

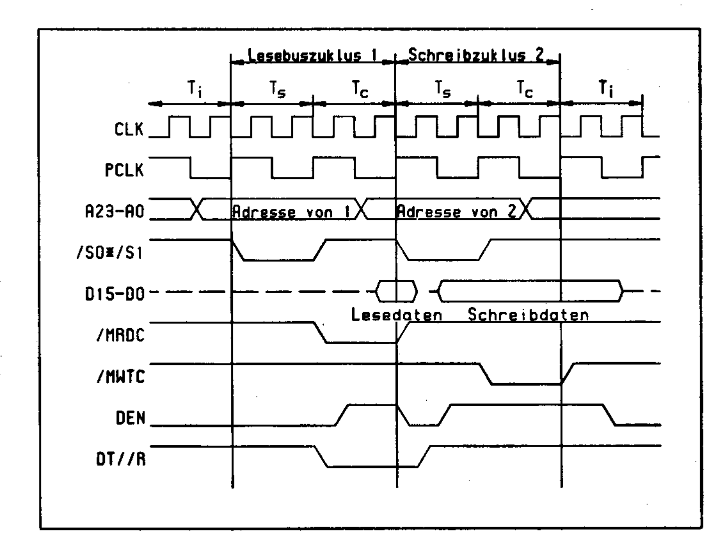


Bild 20 Taktdiagramm des 80286 /8/

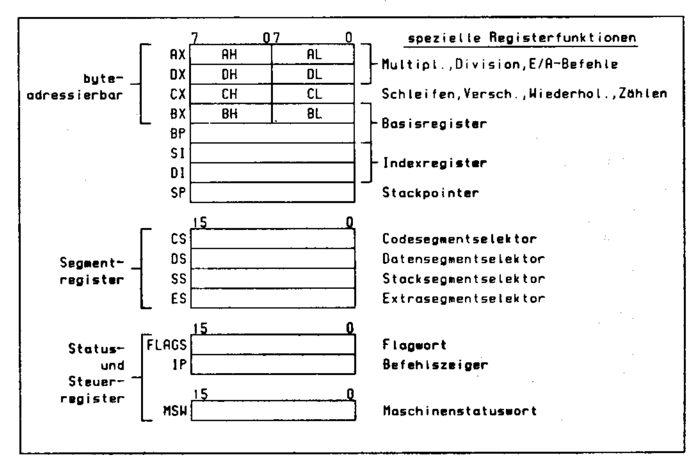


Bild 22 Registersatz des 80286 /8/

Weiterhin gibt es auch beim 80286 die Möglichkeit von Softwareinterrupts und einige der Interrupts sind für Befehlsausnahmen reserviert. Neben den Befehlsausnahmen des 8086 und 80186 gibt es beim 80286 einige zusätzliche, die in Tafel 5 aufgeführt sind.

4.2. Die CPU 80286

Gegenüber dem 8086 wurden beim 80286 besonders die Architektur und die Adressierung verbessert. Geringfügige Veränderungen erfolgten auch am Befehlssatz.

4.2.1. Architektur

In der Architektur des 80286 erfolgten wesentliche Verbesserungen gegenüber dem 8086.
Man kann intern vier selbständige Funktionsblöcke unterscheiden, die weitgehend parallel arbeiten und durch ein inneres Bussystem miteinander in Verbindung stehen (Bild 21).
Als erster Funktionsblock ist da die Buseinheit zu nennen, die beim 8086 schon in ähnlicher Form existierte. Die Buseinheit steuert alle Zugriffe des Prozessors auf den externen Bus. Dabei bestehen Prioritäten zwischen verschiedenen internen und externen Busanforderungen. Falls der externe Bus nicht durch andere Aktivitäten belegt ist, füllt die Buseinheit die sechs Byte lange Codewarteschlange mit Befehlen auf.
Der zweite Funktionsblock im 80286 ist die Befehlseinheit. Diese Einheit holt die Bytes aus der Codewarteschlange, faßt sie zu Befehlen zusammen und decodiert die Befehle. Außerdem werden die Befehle noch vorverarbeitet und in einer drei Einträge langen Befehlswarteschlange abgelegt.
Als weiterer Funktionsblock wäre die Ausführungseinheit zu nennen. Hier erfolgt die eigentliche Abarbeitung der Befehle, die sich in der Befehlswarteschlange befinden. Dazu enthält die Ausführungseinheit eine ALU, den Registersatz und einen Mikrocode-ROM mit insgesamt 1536 Worten zu je 35 Bit.
Der letzte Funktionsblock im 80286 ist schließlich die Adreßeinheit. Die Adreßeinheit übernimmt alle Adreßrechnungen im 80286. Sie berechnet zum Beispiel den Offset, vergleicht ihn mit der Segmentlänge, prüft die Zugriffsrechte und bildet schließlich die physische Adresse, die an den Adreßpins erscheint.
Das interne Bussystem, das alle Einheiten im 80286 verbindet, ist 16 Bit breit.

Tafel 5 Zusätzliche Befehlsausnahmen des 80286 /8/

Int.-Nummer	Funktion
7	Koprozessor nicht vorhanden
8	Doppelfehler
9	Koprozessor-Segmentüberlauf
10	ungültiges Taskzustandssegment (TSS)
11	Segment nicht präsent
12	Stacksegmentfehler
13	allgemeine Privilegverletzung
16	Koprozessorfehler

4.2.2. Registersatz

Der allgemeine Registersatz des 80286 entspricht dem des 8086 (Bild 22).
Er besteht aus acht Registern von 16 Bit Breite, die sowohl Daten als auch Adressen enthalten können. Einige Register haben noch spezielle Funktionen. Befehlszeiger, Segmentregister und Flagregister sind ebenfalls identisch, nur wurden beim 80286 einige der beim 8086 unbenutzten Flags belegt. Diese Flags haben im einzelnen folgende Bedeutung:
– **NT**: Nested Task Flag; steuert Funktion des IRET-Befehls, bei NT = 1 wird Taskumschaltung ausgelöst
– **IOPL**: E/A-Privilegniveau; legt Privilegniveau für E/A-Befehle fest.
Ein zusätzliches 16-Bit-Register ist das Maschinenstatuswort (MSW), das vorwiegend der Zusammenarbeit mit Koprozessoren dient. Weiterhin gibt es beim 80286 eine ganze Anzahl Systemadreßregister, auf deren Aufgaben in Zusammenhang mit der Adressierung eingegangen wird.

4.2.3. Adressierung

Bei der Adressierung des 80286 muß man zwei verschiedene Betriebsarten unterscheiden. Die eine Betriebsart ist der Real-Adreß-Mode. In dieser Betriebsart arbeitet der 80286 als schneller 8086 und ist vollständig objektcodekompatibel, so daß hier auf diese Betriebsart nicht weiter eingegangen wird.
Die Hauptbetriebsart des 80286 ist der Protected-Mode, bei dem erst alle Vorzüge des Prozessors zur Geltung kommen. Darum soll diese Betriebsart auch im Mittelpunkt der weiteren Betrachtungen stehen. Der Übergang vom Real-Adreß-Mode in den Protec-

ted-Mode erfolgt mit einem speziellen Befehl, eine Rückkehr ist nur durch Reset möglich. Im Protected-Mode wird mit virtuellen Adressen, Speicherverwaltung und Speicherschutz gearbeitet. Der physische Adreßbereich kann maximal 16 MByte groß sein, der virtuelle ein GByte. Der Speicher ist in Segmente unterteilt, die zwischen einem Byte und 64 Byte groß sein können.

4.2.3.1. Adressierungsarten

Die logischen Adressierungsarten, über die der Programmierer verfügen kann, sind beim 80286 identisch zum 8086. Es gibt also genauso die Möglichkeiten der Basisadressierung, Indexadressierung und Basis-Indexadressierung im Protected-Mode. Das bedeutet, daß der Prozessor im Protected-Mode quellcodekompatibel für 8086-Software ist.

4.2.3.2. Wandlung der logischen in die physische Adresse

Die Bildung der physischen Adresse aus der logischen erfolgt im Protected-Mode des 80286 auf der Grundlage der Speichersegmentierung. Jedem Segment ist dabei ein Segmentdeskriptor zugeordnet, der bestimmte Informationen über das Segment enthält (Bild 23).
Im einzelnen gibt der Deskriptor Auskunft über Typ, Privilegniveau, Anfangsadresse und Länge des Segments. Die physische Adresse wird gebildet, indem zu der Segmentanfangsadresse aus dem Deskriptor der Offset hinzuaddiert wird. Der gewünschte Segmentdeskriptor wird durch den Segmentselektor, der sich in einem Segmentregister befindet, aus einer Deskriptortabelle im Speicher ausgewählt. Bild 24 zeigt die Wandlung der logischen in die physische Adresse.
Von den Deskriptortabellen gibt es insgesamt drei Arten:

- die globale Deskriptortabelle (GDT),
- die Interruptdeskriptortabelle (IDT) und
- die lokale Deskriptortabelle (LDT).

Die globale Deskriptortabelle enthält die Deskriptoren aller derjenigen Segmente, die allen Tasks zugeordnet sind. Ihre Anfangsadresse und Länge enthält ein Systemadreßregister (GDTR), weshalb sie beliebig im Speicher lokalisiert werden kann. Sie kann maximal 8192 Deskriptoren enthalten.
Die Interruptdeskriptortabelle bestimmt die Startadressen der bis zu 256 Interruptbehandlungsroutinen. Der gewünschte Deskriptor wird durch den 8-Bit-Interruptvektor ausgewählt. Anfangsadresse und Länge der IDT sind wiederum in einem Systemadreßregister (IDTR) enthalten.
Die lokale Deskriptortabelle enthält die Deskriptoren der Segmente, die nur einer Task zugeordnet sind. Jede Task besitzt eine eigene LDT, die erst beim Start der Task geladen werden muß. Jede LDT ist ein Segment, dem ein LDT-Deskriptor zugeordnet ist. Dieser LDT-Deskriptor muß in der GDT enthalten sein. Ein Systemadreßregister (LDTR) enthält immer den Selektor, der auf den aktuellen LDT-Deskriptor in der GDT verweist. Beim Laden dieses Registers werden Anfangsadresse und Länge der LDT in Cacheregister geholt.
Zur Beschleunigung der Adreßumwandlung sind auch jedem Segmentregister Cacheregister zugeordnet, die beim Wechsel des Segmentregisterinhaltes mit dem Inhalt des zugehörigen Deskriptors geladen werden.

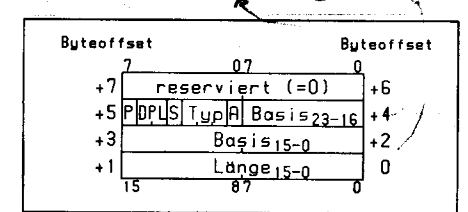


Bild 23 Segmentdeskriptor des 80286 /8/

4.2.3.3. Multitaskingunterstützung

Um ein Multitasking- und Multiusersystem zu implementieren, ist es notwendig, daß der Zielrechner auch über Möglichkeiten verfügt, um Speicherbereiche und Tasks gegeneinander zu isolieren und einen schnellen Taskwechsel vorzunehmen. Der Prozessor 80286 unterstützt dies bereits hardwaremäßig.
Zur Isolation der Task dienen schon die lokalen Deskriptortabellen jeder Task.
Um Segmente vor unerlaubtem Zugriff zu schützen, existieren beim 80286 vier Privilegniveaus, von 0 bis 3 (siehe Bild 25).
Durch diese Einrichtung ist es möglich, Anwenderprogramme und Betriebssystem voneinander zu isolieren. Im Bild ist eine mögliche Verwendung der Privilegniveaus angedeutet. Jedem Segment ist ein Privilegniveau zugeordnet (DPL), enthalten im Segmentdeskriptor. Außerdem besitzt jede Task ein Privilegniveau (CPL), welches durch den Inhalt des CS-Registers bestimmt wird. Das CPL einer Task ist nicht statisch, sondern kann sich ändern. Es gibt nun verschiedene Privilegregeln, bei deren Verletzung eine Befehlsausnahme erfolgt. Innerhalb jeder Task existiert zu jedem Prvilegniveau ein eigener Stack. Der Privilegniveauwechsel einer Task erfolgt über den Gatemechanismus, bei dem feste Einsprungstellen in das höhere Privilegniveau existieren.
Damit eine schnelle Taskumschaltung beim 80286 ermöglicht wird, wird jede Task durch ein Taskzustandssegment (TSS) beschrieben. Dieses TSS enthält die Inhalte aller Register, den LDT-Selektor der Task, Stackpointer für die Privilegniveaus 0,1 und 2 und einen Rückverweis auf eine vorhergehende Task. Das TSS, welches zur gerade laufenden Task gehört, wird durch ein Systemadreßregister (TR), das den entsprechenden Selektor enthält, bestimmt. Eine Taskumschaltung kann durch einen JMP- bzw. CALL-Befehl oder durch Interrupt ausgelöst werden. Der Selektor bzw. Interruptvektor muß dabei auf einen gültigen TSS-Deskriptor oder ein Task-Gate in der entsprechenden Tabelle verweisen. Bei 8 MHz Taktfrequenz dauert eine Taskumschaltung 21 μs /7/. In dieser Zeit werden alle Registerinhalte gewechselt, die benötigten Deskriptoren in Caches geladen, auf die neue LDT umgeschaltet und in das neue TSS ein Rückverweis auf das alte eingetragen.

4.2.4. Befehlssatz

Im Befehlssatz des 80286 sind alle Befehle des 8086 und 80186 enthalten. Daneben existieren einige neue Befehle, um die Systemadreßregister und das MSW zu laden. Andere neue Befehle sind zur Prüfung von Zugriffsrechten eingeführt worden.
Weiterhin wurden beim 80286 viele Befehle abhängig vom Privilegniveau der Task gemacht und werden nur unter bestimmten Bedingungen ausgeführt. Das betrifft besonders Befehle zur Prozessorsteuerung. E/A-Befehle sind vom I/O-Privilegniveau abhängig, welches durch das Flagregister bestimmt wird.

4.3. Koprozessoren

Um die Leistungsfähigkeit eines 80286-Systems noch weiter zu erhöhen, ist es möglich, die CPU durch Koprozessoren zu ergänzen. Diese Koprozessoren sind auf bestimmte Aufgaben spezialisiert und entlasten den Hauptprozessor.
Hier eine Auswahl von Koprozessoren
- **80287**: Arithmetikprozessor, spezialisiert auf Gleitkomma Arithmetik, algebraische und transzendente Funktionen
- **82258**: Advanced-DMA-Prozessor mit vier DMA-Kanälen, einer davon als Multiplexkanal mit 32 Unterkanälen, Übertragungsrate bis 8 MByte/s, speziell für E/A-Operationen einsetzbar
- **82786**: Grafikprozessor für Bit-Map-Grafik, verwaltet eigenen Bildwiederholspeicher, unterstützt Fenstertechnik, leistungsfähige Zeichenbefehle
- **82730**: Textprozessor, spezialisiert auf Textverarbeitung am Bildschirm
- **82586**: LAN-Prozessor für Ethernet mit maximal 10 MBit/s, realisiert das komplette CSMA/CD-Protokoll für den Zugriff auf das Übertragungsmedium
- **82588**: LAN-Prozessor für Netze mit maximal 2 Mbit/s

Auf die beiden ersten Koprozessoren wird im folgenden noch näher eingegangen. Bis auf den Arithmetikprozessor können alle anderen Koprozessoren auch unabhängig (remote) betrieben werden. Die Verbindung zur CPU erfolgt dann über den Systembus.

4.3.1. Arithmetikprozessor

Der Arithmetikprozessor 80287 arbeitet parallel zur CPU. Notwendige Operandentransfers werden allerdings durch die Buseinheit des 80286 durchgeführt, um das Schutzkonzept nicht zu durchbrechen. Die CPU verfügt deshalb über ein spezielles Interface zum Arithmetikprozessor.
Softwareseitig ist der 80287 kompatibel zum 8087, so daß existierende Programme weiterverwendet werden können. Der 80287 besteht intern aus zwei Einheiten, der Busanschlußeinheit (BIU) und der Numerikausführungseinheit (NEU), die über ein inneres Bussystem mit 80 Bit Breite verbunden sind (siehe Bild 26).
Beide Einheiten arbeiten teilweise parallel. Die NEU enthält einen Registersatz mit acht Gleitkommaregistern von 80 Bit Breite, der als Stack organisiert ist.

4.3.2. ADMA-Prozessor

Der ADMA-Prozessor 82258 besitzt Leistungsmerkmale, die bei älteren DMA-Controllern nicht vorhanden waren. Das lokale Businterface des 82258 läßt sich für die ganze 8086-Familie konfigurieren. Er ist also problemlos mit dem 8086 im Minimum- oder Maximummode, mit dem 80186 und dem 80286 betreibbar. Daneben gibt es auch die Möglichkeit, den ADMA-Prozessor unabhängig zu betreiben.

4.3.2.1. Arbeitsmodi

Beim 82258 kann man grundsätzlich zwischen zwei Arbeitsmodi unterscheiden. Der erste Mode ist die lokale Betriebsart. Diese Betriebsart ist in Bild 27 dargestellt.
In dieser Betriebsart teilt sich der 82258 den lokalen Bus mit der CPU. Er nutzt den Bus-

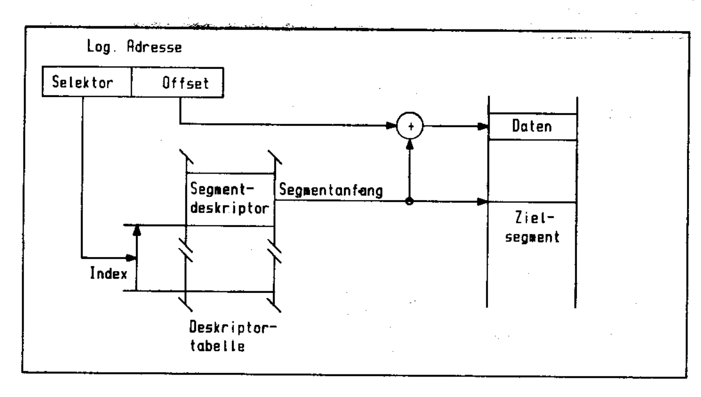


Bild 24 Segmentadressierung im Protected-Mode des 80286 /8/

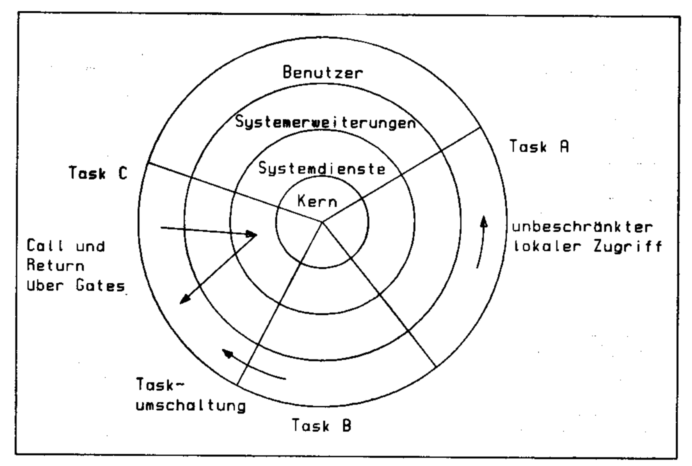


Bild 25 Privilegniveaus im Protected-Mode des 80286 /9/

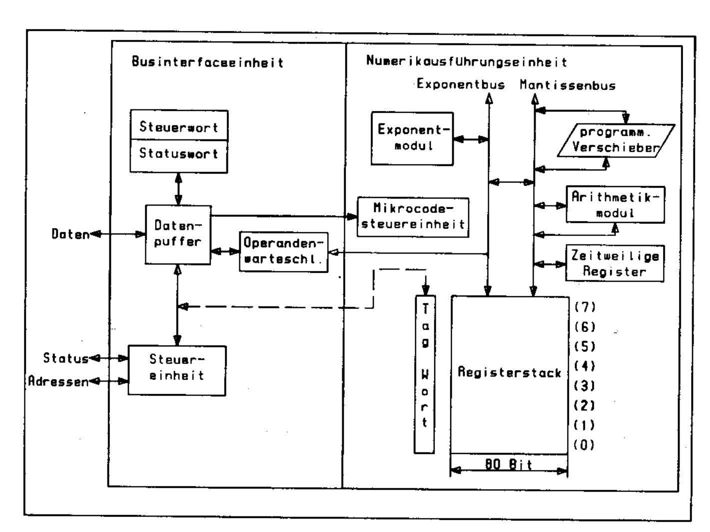


Bild 26 Blockschaltbild des 80287 /16/

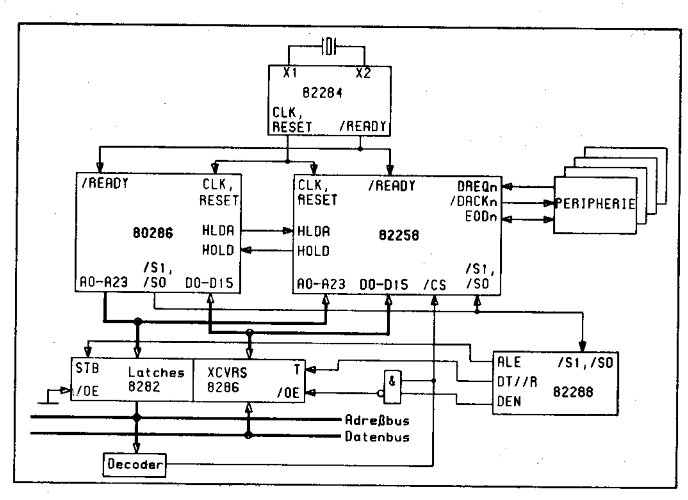


Bild 27 ADMA-Prozessor 82258 in der lokalen Betriebsart /17/

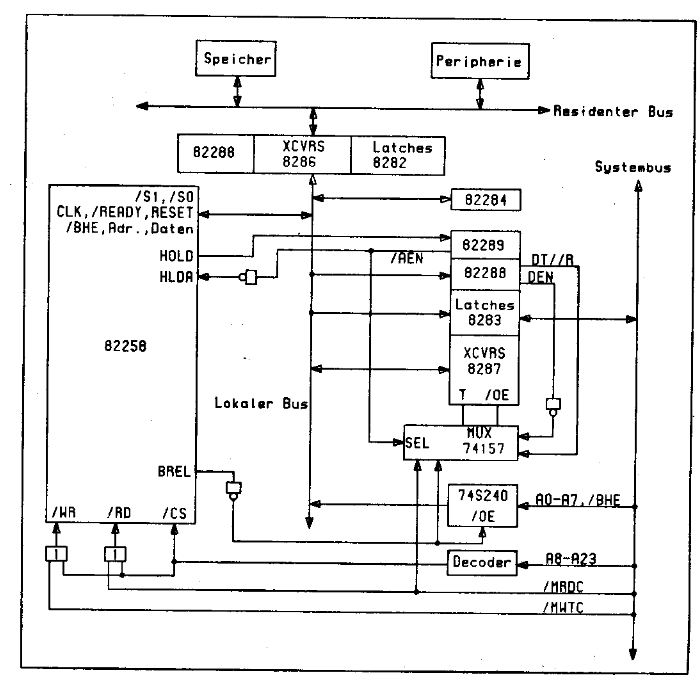


Bild 28 ADMA-Prozessor 82258 im unabhängigen Betrieb /17/

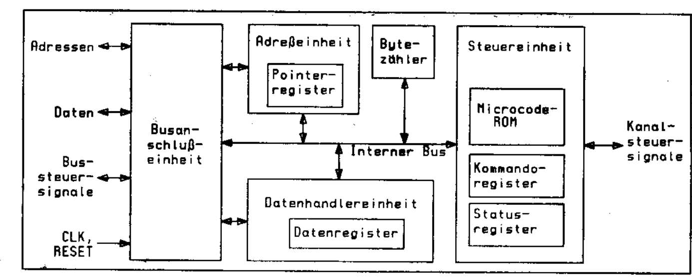


Bild 29 Blockschaltbild des ADMA-Prozessors 82258 /17/ ▶

controller, die Adreßlatches und Datenbustreiber der CPU, so daß kein zusätzlicher Aufwand entsteht. Die Vergabe des lokalen Busses erfolgt über den HOLD/HLDA-Mechanismus. Da der 82258 über Burst- und Delay-Counter verfügt, lassen sich Frequenz und Dauer seiner Buszugriffe optimal einstellen. Die CPU kann auf die Register des 82258 synchron zugreifen. Der 82258 kann in der lokalen Betriebsart 16 MByte Speicheradreßraum und zusätzlich 16 MByte E/A-Adreßraum adressieren.

Die zweite Betriebsart ist der unabhängige Mode (Bild 28). Dabei besitzt der 82258 ein eigenes residentes Bussystem und kommuniziert mit der CPU über den Systembus, z. B. den Multibus. Die CPU kann aber den Systembus auch asynchron auf die Register des 82258 zugreifen. Der 82258 kann im unabhängigen Mode 16 MByte am residenten Bus und 16 MByte am Systembus adressieren. In dieser Betriebsart entsteht allerdings ein beträchtlicher zusätzlicher Aufwand, da der 82258 über eigene Buscontroller, Adreßlatches und Datenbustreiber verfügt. Allerdings läßt sich damit ein modulares E/A-Subsystem aufbauen, das parallel zur CPU arbeitet.

4.3.2.2. Leistungsmerkmale

Das Blockschaltbild des 82258 zeigt Bild 29. In diesem Blockschaltbild sind fünf interne Funktionsblöcke erkennbar. Die Busanschlußeinheit gewährleistet die Verbindung nach außen. Die Adressen für jeden Buszyklus werden durch die Adreßeinheit bereitgestellt. Die Datenhandlereinheit hat die Aufgabe, im bearbeiteten Datenstrom nach Matchbytes zu suchen oder einen Vergleich von Daten vorzunehmen. Der Bytezähler zählt die übertragenen Bytes, um das Blockende zu erkennen. Die Steuereinheit schließlich koordiniert alle Abläufe im 82258. Sie enthält dazu den Mikrocode-ROM mit dem Mikroprogramm für die einzelnen Kommandos.

Der 82258 ist in der Lage, Kanalprogramme, die im Speicher stehen, abzuarbeiten. Diese Kanalprogramme können auch bedingte Relativsprünge oder bedingte Stopps enthalten. An speziellen DMA-Operationen sind das

Füllen, Übersetzen, Vergleichen von Datenblöcken oder das Suchen nach bestimmten Bitmustern möglich. Weiterhin können Bytes zu Wörtern zusammengesetzt werden und umgekehrt. Es ist auch möglich, Datenblöcke zusammenzusetzen oder auf verschiedene Ziele zu verteilen.
Zwischen den vier Kanälen sind verschiedene Prioritätsmodi programmierbar. Die Umschaltung zwischen den Kanälen erfolgt ohne Verlustzeiten. Ein Kanal kann als Multiplexkanal mit maximal 32 Unterkanälen betrieben werden, wobei der Multiplexer durch Interruptcontroller 8259A realisiert wird.

wird fortgesetzt

Literatur

/6/ Zingale, T.: Intel's 80186, a 16-bit Computer on a Chip. BYTE (1983) 4
/7/ Wells, P.: The 80286 Microprocessor BYTE (1984) 11
/8/ iAPX 286/10 High Performance Microprocessor with Memory Management and Protection. Intel Corporation 1982
/9/ Geyer, J.: Schneller Prozessor arbeitet mit virtuellem Adreßbereich Elektronik 19 (1982) 5
/10/ Childs; Crawford; House; Noyce: A Processor Family for Personal Computers. Proceedings of the IEEE Vol. 72/März 1984
/11/ 82284 Clock Generator and Ready Interface for iAPX 286 Processors. Intel Corporation 1982
/12/ 82288 Bus Controller for iAPX 286 Processor. Intel Corporation 1982
/13/ 82289 Bus Arbiter för iAPX 286 Processor Family. Intel Corporation 1984
/14/ Krelle, D.: Mehr Systemleistung mit Coprozessoren. Elektronik 22 (1985) 23
/15/ Böning, W.: 16-Bit-DMA-Controller bedient schnelle Peripherie. Elektronik 23 (1986) 9
/16/ 80287 80-bit HMOS Numeric Processor Extension. Intel Corporation 1982
/17/ 82258 Advanced Direct Memory Access Coprocessor (ADMA). Intel Corporation 1985

TURBO-TIPS

Rückkehr ins aufrufende Programm

Steffen Roller

Um aus einem TURBO-Pascal-Programm heraus ein COM-File aufzurufen, steht die Standard-Procedure EXECUTE zur Verfügung. Nach Abarbeiten der COM-Datei kehrt man jedoch nicht ohne weiteres in das aufrufende Programm zurück. Diese kleine Routine in TURBO-Pascal schreibt eine Datei mit Namen $$$.SUB, die die weitere Arbeit für uns übernimmt.
Nichts anderes macht übrigens das SCP-Kommando SUBM.
Wichtig ist, die SCP-Kommandos in UMGEKEHRTER Reihenfolge ihrer Abarbeitung in die $$$.SUB-Datei zu schreiben. Während der Abarbeitung löscht das Betriebssystem jedes ausgeführte Kommando in der Datei, so daß sie nach dem letzten ausgeführten Kommando verschwunden ist.

```
PROCEDURE schreibe(kommando:s15typ);
    VAR i      : Integer;
        puffer : ARRAY [0..127] OF Byte;

    BEGIN
        puffer[0]:=Length(kommando);
        FOR i:=1 TO Length(kommando) DO
              puffer[I]:=Ord(kommando[i]);
        puffer[i+1]:=0;
        BlockWrite(sub,puffer,1);
    END;

(* Beispiel fuer die Anwendung der PROCEDURE :

PROGRAM prog1;
   TYPE s15typ = STRING[15];
        fo     = FILE;
   VAR  sub    : fo;

   {$I SUBWRITE.INC }                { Einbinden als Include-File         }

   BEGIN
      Assign(sub,'$$$.SUB');
      Rewrite(sub);
      schreibe('PROG1');            { zuerst das LETZTE Kommando schreiben }
      schreibe('PROG2');            { danach das vorhergehende usw.        }
      Close(sub);
      Halt                          { verlassen von PROG1.COM mit Warmstart}
   END.                             { $$$.SUB uebernimmt die Steuerung     }
*)
```

COM 2: unter TURBO-PASCAL

Christian Hanisch

Serielle Schnittstellen werden auf IBM-kompatiblen PCs unter MS-DOS-kompatiblen Betriebssystemen häufig zum Anschluß von Geräten wie Plotter, Digitizer, Modem etc. genutzt.
TURBO-PASCAL unterstützt durch das AUX:-Gerät nur die COM1:-Schnittstelle. Wenn mehrere serielle Schnittstellen belegt sind und aus TURBO heraus angesprochen werden sollen, so kann dies mit einem sehr einfachen benutzergeschriebenen I/O-Driver realisiert werden.
Dieser Driver in Form der im Bild 1 dargestellten Prozedur COM2 nutzt den **Interrupt 14H** (serielle Schnittstelle) des MS-DOS-Betriebssystems. Zum Zwecke des Sendens von Daten wird das Register AH auf 1 gesetzt, das zu sendende Zeichen nach AL gebracht, im DX-Register die konkrete Schnittstelle spezifiziert (0 – COM1:, 1 – COM2:, ...) und der Interrupt 14H durch die INTR-Prozedur von Turbo aufgerufen.
Im Hauptprogramm ist der **UsrOutPtr** auf die Adresse der Prozedur COM2 „umzubiegen" (Bild 1: Zeile 16). Die Kommunikation erfolgt dann problemlos über das USR:-Gerät (Bild 1: Zeile 18).

```
 1: 0 PROGRAM Demo_COM2_Handling;
 2: 0
 3: 0 PROCEDURE com2(ch:CHAR);
 4: 0 VAR reg: RECORD CASE INTEGER OF
 5: 1             1: (ax,bx,cx,dx,bp,si,di,ds,es,flags:INTEGER);
 6: 1             2: (al,ah,bl,bh,cl,ch,dl,dh: BYTE);
 7: 1            END;
 8: 0 BEGIN
 9: 1   reg.ah:=1;             (* 1 = Zeichen Senden          *)
10: 1   reg.al:=BYTE(ch);      (* Zeichen nach AL             *)
11: 1   reg.dx:=1;             (* 1 = Schnittstelle COM2:     *)
12: 1   intr(20,reg);          (* Interrupt 14H aufrufen      *)
13: 1 END;
14: 0
15: 0 BEGIN
16: 1   usroutptr:=ofs(com2);
17: 1   (* z. B. Plotter bedienen *)
18: 1   write(usr,#13#13#13#13#13,':');
      :
      :
xx: 1 END.
```

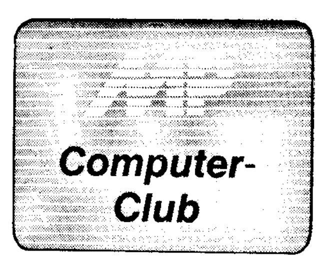
Bild 1: Prozedur COM2

In eigener Sache

Für eine weitere Redakteurstelle in unserer Zeitschrift suchen wir einen geeigneten Mitarbeiter bzw. Mitarbeiterin mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik.
Zu den Aufgaben gehören unter anderem das redaktionelle und fachliche Bearbeiten von Manuskripten, der Besuch und die Auswertung von Fachtagungen, -messen und -ausstellungen, die Zusemmenarbeit mit Autoren und Gutachtern sowie ggf. das Testen und Beurteilen von Programmen, die der Redaktion zur Veröffentlichung eingereicht werden.

Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, rufen Sie uns unter Tel. 2870371 oder 2870203 an oder schreiben Sie an:

VEB Verlag Technik, Redaktion MP, Oranienburger Str. 13/14, Berlin, 1020



Z-1013-Tastatur mit Raffinessen

Rainer Brosig
Informatikzentrum des Hochschulwesens an der TU-Dresden, IG Heimcomputer

Vorbetrachtung

In zahlloser Folge sind in letzter Zeit Tastaturvarianten für den Mikrorechner Z 1013 in der Literatur vorgestellt worden, die sich grob in vier Gruppen teilen lassen:

1. Verbesserung der Bedienbarkeit durch bessere Schaltelemente
2. Abfragen einer größeren Matrix (z. B. K 7659, 8 × 12) und Umsetzen in die originale 4 × 8-Matrix (z. B. Prozessorversion v. Robotron-Elektronik Riesa)
3. Verzicht auf eine Matrixschnittstelle und Ersatz durch eine parallele oder serielle ASCII-Schnittstelle
4. Änderung des Monitor-Tastaturprogramms, um eine komfortablere Abfrage mit einer veränderten Matrix zu realisieren. Alle vier Gruppen haben ihre speziellen Vor- und Nachteile. Es kann festgestellt werden, daß es keine ideale Lösung für alle Nutzer gibt, sondern daß alle vier Gruppen ihren speziellen Nutzerkreis haben.

Die erste Gruppe von Tastaturen wird z. B. besonders die Nutzer interessieren, die noch wenig mit der Computermaterie vertraut sind, da nur allgemeine elektronische Kenntnisse notwendig sind. Für fortgeschrittene Nutzer soll folgender Tastaturvorschlag gedacht sein, der sich in die vierte Gruppe einordnet, für die es bisher wenig Veröffentlichungen gab.

Lösungsprinzip der Tastaturvariante

Es wird eine K 7659 mit einer speziellen Hardwareanpassung verwendet, die an den gleichen Hardwareschnittstellen wie die Originaltastatur betrieben wird (Bild 1 bis Bild 3). Dadurch gibt es wenig Probleme bei Software, die direkt auf die Originalmatrix zugreift. Die Tastaturabfrage erfolgt durch ein neues komfortables Programm, welches verständlicherweise länger als das originale ist und deshalb außerhalb des 2-K-Monitors generiert wird (ab 0F800H). Um Softwarekompatibilität zu sichern, wurde am Anfang der originalen Tastaturroutine ein Sprung zum externen Tastaturprogramm eingetragen. Dazu ist es erforderlich, den originalen Monitor zu ändern. Auf den durch den Wegfall der Originalroutine frei gewordenen Speicherplatz im Monitor wurden kleine Hilfsroutinen, wie die Ausgabe auf einen Port (0) und die Anzeige des Kommandoverteilers auf 0B0H (Z), gelegt. Weiterhin befindet sich dort eine NMI-Unterbrechungsroutine, wo, mit Hilfe einer NMI-Taste, ein Maschinenprogramm mit Registeranzeige abgebrochen werden kann, was sich bisher als eine außerordentliche Hilfe beim Test von Assemblerprogrammen erwiesen hat.

Zur Änderung des Monitor-EPROMs muß nicht unbedingt ein Eingriff auf der originalen Leiterkarte erfolgen, was evtl. Garantieansprüche löschen könnte. Es besteht die Möglichkeit, den Z 1013 im Originalzustand zu belassen und den geänderten Monitor-EPROM von außen zuzuschalten (MEMDI-Steuerung). Einfacher ist es aber, den originalen Monitor-EPROM auszulöten und durch einen anderen EPROM mit Fassung zu ersetzen. Derzeit wird für Monitor und Tastatur ein 2732 benutzt. Auf diesem EPROM befinden sich weiterhin das Headersave sowie ein Sprungverteiler für Peripherie-Schnittstellen.

Hardware

Die zusätzliche Hardware beschränkt sich auf einen 1-aus-16-Decoder, die Statussteuerung und einen 8-auf-4-Coder. Zur Anzeige von Shift-lock (CAPS) und Hardcopy werden zwei Anzeige-Flip-Flops verwendet. Damit 12 Spalten abgefragt werden können, muß die 4-Bit-Spalteninformation neu decodiert werden, da der originale Spaltendecoder nur bis 10 decodiert. Dazu ist es erforderlich, daß die 4-Bit-Spalteninformation vom Spaltenlatch A47 über die vier freien Kontakte auf dem Lötkamm des Tastaturanschlusses geführt wird.

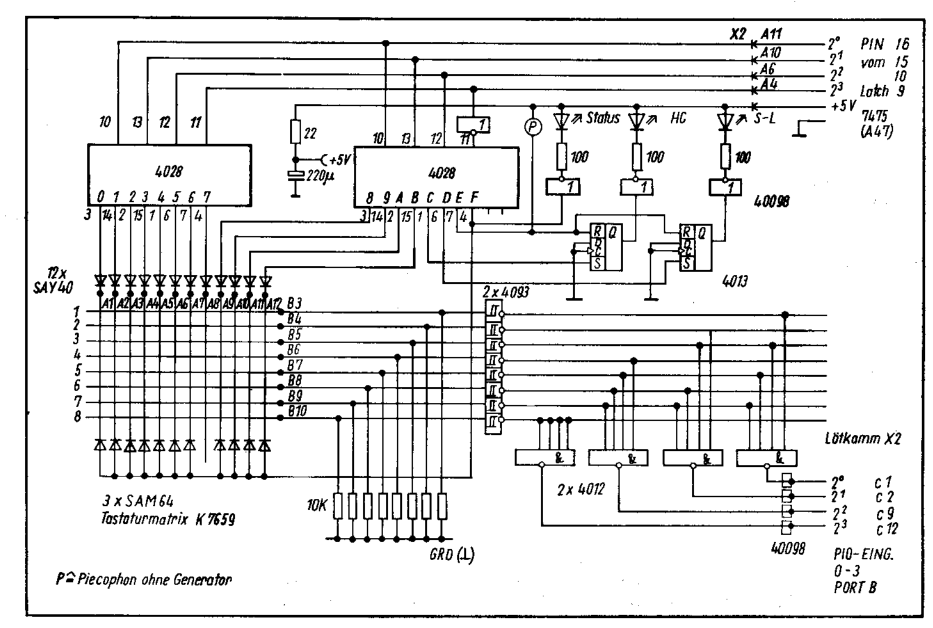
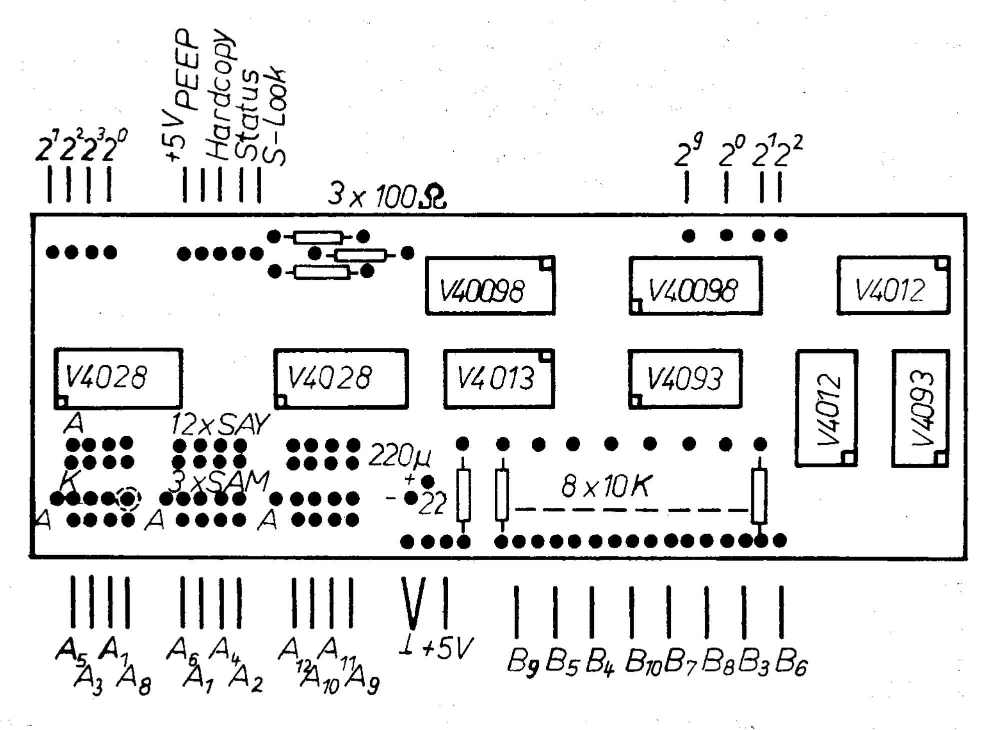


Bild 1 Schaltplan der Leiterplatte
Bild 2 Bestückungsplan der Tastaturleiterplatte



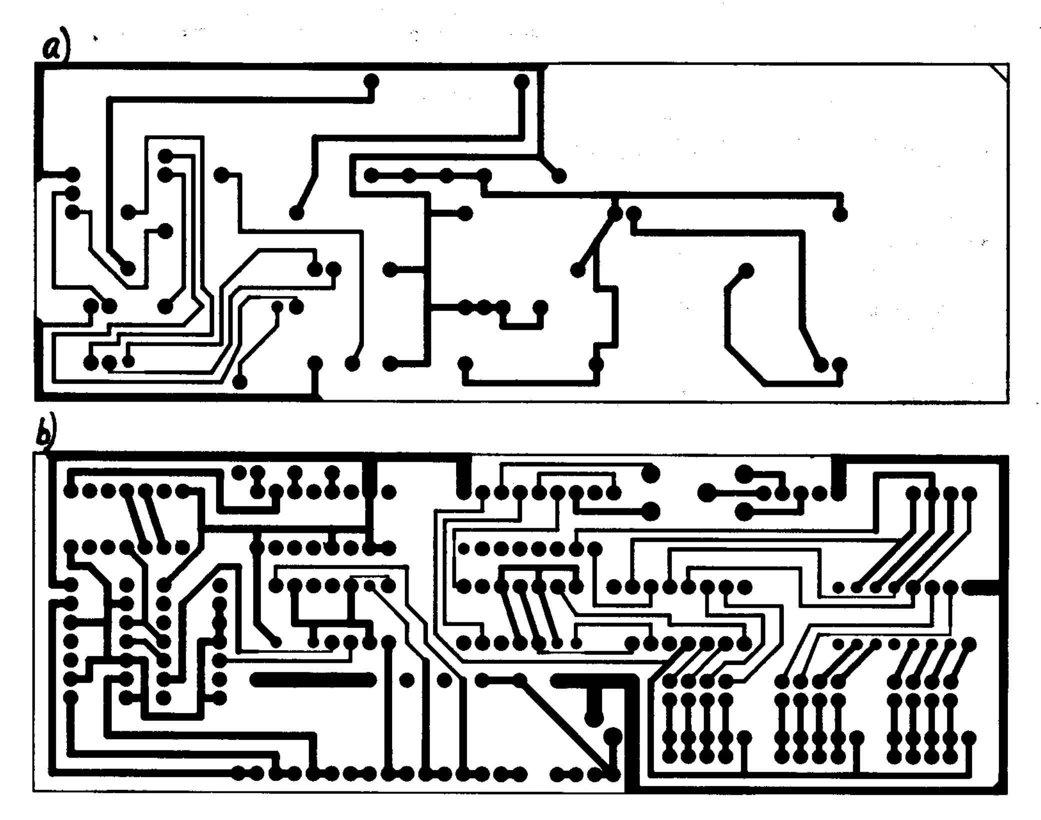


Bild 3a **Layout, B-Seite** *Bild 3b* **Layout, L-Seite**

Sollten auch diese vier Drähte zu viel Änderung auf der Platine darstellen, so kann der Spaltenlatch auch extern über den Systemstecker neu decodiert werden.
Die verbleibenden vier Digits vom 1-aus-16-Decoder werden für die Ansteuerung der Flip-Flops, der Statussteuerung und des Tastenbeeps (generatorloser Piezoschwinger o. ä.) genutzt. Die Ausgänge des Decoders steuern die Spaltenleitungen der Tastaturmatrix, die über Dioden entkoppelt sind. Diese Entkopplung ist notwendig, um bei der Betätigung von zwei Tasten (SHIFT, CTRL) keine Spalten kurzzuschließen. Die Zeilenausgänge werden durch einen Coder auf einen negierten binären Code gewandelt, um wieder die originalen 4-Bit-Zeileneingangsleitungen zu erhalten. Deshalb sind prinzipiell Direktzugriffe auf die Matrix möglich, die allerdings nicht die gleiche Tastenbelegung wie die Original-Tastatur aufweist (Korrespondenztabelle benutzen). An dieser Stelle soll auch darauf hingewiesen werden, daß solche unnötigen Praktiken, direkt auf die Matrix zuzugreifen, immer zu Softwareinkompatibilitäten führen und deshalb zu vermeiden sind.
Das verwendete Abfrageprinzip weist eine Besonderheit auf, die in einer Statussteuerung liegt.
Diese Statussteuerung hat die Aufgabe, alle Spalten der Tastatur aktiv zu schalten, um einerseits eine schnelle Tastaturabfrage zu realisieren (der Rechner muß nicht ständig die Spaltenleitungen abfragen) und andererseits die Interruptfähigkeit zu ermöglichen (wird momentan nicht ausgenutzt). Die Abfrage der Tastatur bei unbetätigten Tasten wird durch die Statussteuerung etwa 100 mal schneller als die originale. Dieser Zeitvorteil hat aber nicht nur positive Seiten. Programme, die die Abarbeitungszeit für bestimmte Zwecke ausnutzen, laufen dadurch zu schnell (z. B. blinkender Cursor). Deshalb kann per Tastendruck für solche Anwendungen eine zusätzliche Zeitschleife eingeschaltet werden.
Die Funktion der Statussteuerung wird mittels LED angezeigt, die damit die Bereitschaft einer Dateneingabe über die Tastatur signalisiert.
Die schaltungstechnische Realisierung kann mit verschiedenen Mitteln erfolgen. Die einfachste Möglichkeit besteht in der Decodierung mit einem 74154, der Codierung mit 2 DL 021 und einem DL 074 für die Anzeige. Somit kann die gesamte Hardware auf 4 ICs beschränkt bleiben. Um aber das arg strapazierte Netzteil nicht noch mehr zu belasten, wurde eine etwas aufwendigere CMOS-Schaltung entwickelt, die dazu noch die höhere Sicherheit gegen unerwünschte Übergangswiderstände aufweist. Die Spalte 7 (SHIFT, CTRL) wird durch die Statussteuerung nicht aktiviert (Fehlen einer Diode), damit beim alleinigen Drücken von SHIFT oder CTRL der Tastaturstatus nicht verändert wird. Diese Spalte wird erst nach Erkennung einer anderen Taste abgefragt, um die SHIFT-Ebene umzuschalten bzw. eine Control-Maske (1FH) aufzulegen. Die geringe Anzahl zur Verfügung stehender zusätzlicher Steuerleitungen macht es erforderlich, für die Anzeige der Funktionen SHIFT-LOCK und Hardcopy zusätzliche Flip-Flops mit einer gemeinsamen Resetleitung zu verwenden. Bei jeder Neubelegung eines Flip-Flops werden deshalb durch den Rechner immer beide Flip-Flops neu gesetzt. An der Resetleitung ist ebenfalls der Tastenklick angeschlossen, wodurch die Flip-Flops auch nach der Ausgabe des Tones neu gesetzt werden müssen. Der Tastenklick wird softwaremäßig erzeugt und ist durch Tastendruck ein- und ausschaltbar.
Für die CMOS-Version wurde eine Leiterplatte entwickelt und digitalisiert, die von der Größe her dem Raum zwischen dem Haupttastenfeld und den rechten Funktionstasten angepaßt wurde. Damit muß die Leiterplatte nicht unterhalb der Tastatur montiert werden, und es ergibt sich eine sehr flache moderne Bauform für das Tastaturgehäuse. Das moderne Äußere wird auch durch das nur 10adrige dünne Kabel von der Tastatur zum Rechner unterstützt, für das 10adrige flexible Telefonleitung verwendet wurde.

Software

Für die Tastaturabfrage wurde ein neues Abfrageprogramm entwickelt, welches der verwendeten Hardware angepaßt ist. Es hat mit Tabellen eine Länge von einem 3/4 KByte und kann deshalb nicht den originalen Standort beziehen. Durch den auf der Originalaufrufadresse (F130H) eingetragenen Sprung zum neuen Standort (F800H) wird die Software-Kompatibilität zum Originalmonitor erreicht. Es muß aber eingeschränkt werden, daß Programme, die in die originale Tastaturroutine springen (eine laienhafte Unsitte), nicht mehr fehlerfrei laufen.
Bei der Realisierung des Programms wurde ein in jeder Hinsicht offenes Prinzip verwendet. Das heißt, daß der Nutzer in der Lage ist, sich die Tastatur nach eigenen Wünschen zusammenzustellen. Bis auf zwei Tasten (SHIFT, CTRL) sind alle Tasten neu belegbar. Weiterhin kann auch wahlweise ein String auf beliebige Tasten gelegt werden. Wenn dafür keine besonderen Tasten verwendet werden sollen, können Strings auch durch gemeinsames Drücken von SHIFT und CTRL sowie einer dritten Zeichentaste erzeugt werden. Der Tastaturtreiber erzeugt dazu einen Grafikcode (> 80H), mit dem ein mit diesem Zeichen versehener String in einem Stringfeld gesucht und ausgegeben wird. Sollte kein String mit diesem Kennzeichen existieren, wird dieses interne Grafikzeichen ausgegeben.
Die Ausgabe von Grafikzeichen im Grafikmode wird dadurch nicht beeinflußt.
Für die einzelnen Felder (Stringfeld, Tastencodefeld, Funktionstastenfeld und Funktionstastenadreßfeld) gibt es ein Pointerfeld, welches auf dem ehemaligen Tastencodefeld ab 3BH angelegt wurde. Damit ist es dem Anwender möglich, sich durch das Verändern dieser Pointer an beliebiger Stelle im RAM ein neues Feld aufzubauen.
Diese Möglichkeit hat besonders für das Stringfeld große Bedeutung, da man dadurch in der Lage ist, Strings für verschiedene Anwendungen zu laden (z. B. BASIC-Schlüsselwörter, Assemblermnemonik, Brieffloskeln u. ä.). Durch die Möglichkeit, auch die anderen Felder neu zusammenzustellen, kann auch dem Problem begegnet werden, daß z. B. Programme verschiedene Tastencodes für gleiche Steuerfunktionen benutzen.
Das Pointerfeld wird durch den Monitor beim ersten Reset auf die Standardwerte gestellt (nochmaliges Reset wird am C3H auf Adr. 66H erkannt), womit gesichert ist, daß bei jedem weiteren Reset ein evtl. geändertes Pointerfeld erhalten bleibt. Das hat aber die Konsequenz, daß ein Programmabsturz auch dieses Pointerfeld ungewollt zerstören kann und der Rechner trotz Reset nicht mehr bedienbar ist. Dann hilft nur noch Ausschalten des Rechners! Ändert man folgende vier Zellen, so kann erreicht werden, daß die Neuinitialisierung bei jedem Reset erfolgt.

F02AH 02H → F2H
F02BH F2H → F1H
F02DH 43H → 33H
F030H 0AH → 1AH

Man sollte dann aber auch darauf achten, daß aus Programmen nicht mit Reset oder Sprung zu F000H herausgegangen wird, sondern mit der Monitorrückkehrtaste (RST 38H). Ansonsten muß die Pointerinitialisierung neu erfolgen.
Für die einzelnen Modes der Tastatur (Grafik, Shift-lock, Hardcopy, Stringmode, Beep, Slow/Fast) existieren Flags, die in der Systemzelle 27H stehen (ehemaliger Schalter Grafik/Alpha).
Ein gesetztes Flag (1) bedeutet darin:

Bit 0 – lange Repeatverzögerung durchlaufen
Bit 1 – Stringmode on
Bit 2 – Grafikmode on
Bit 3 – Hardcopy on
Bit 4 – Shift-lock (gilt nur für Buchstaben)
Bit 5 – Tastenklick on
Bit 6 – Slow
Bit 7 – Taste war betätigt.

Diese Flags werden durch den Tastaturtreiber gesetzt bzw. rückgesetzt. Indem man diese Flags anderweitig setzt oder rücksetzt, kann die Betriebsart der Tastatur von außen manipuliert werden.
Bei den Funktionstasten fallen fünf Tasten besonders auf. Mit der Taste F01 läßt sich aus einem beliebigen Anwenderprogramm heraus zum Monitor (RST 38H) zurückkehren, sofern die Tastatur noch abgefragt wird. Dabei wird auch gleichzeitig eine Reininitialisierung des Zusatz-Kommandoverteilers auf B0H durchgeführt, so daß die standardmäßig eingetragenen Kommandos (@L, @S, @D, @I) wieder verfügbar werden.
Die Tasten F06, F07, F08 dienen für

Funktionstasten

(erzeugen keinen Code):

- F01 – RST 38H
- F06 – JMP 100H
- F07 – JMP 200H
- F08 – JMP 300H
- F09 – SCREEN-COPY
- F53 – GRAFIK ON/OFF
- E53 – HARDCOPY ON/OFF
- D53 – SLOW/FAST
- C53 – BEEP ON/OFF
- D00 – SHIFT-LOCK ON/OFF
- B11 – SHIFT (+B99)
- C00 – CTRL

Steuerzeichentasten (derzeitige Belegung)

Pos.	1.Code	2.Code	1.Funktion	2.Fkt.(m.Shift)	1.CTRL	2.CTRL
A00	0AH	0AH	Cursor runter	Cursor runter	^J	^J
A01	0BH	0BH	Cursor hoch	Cursor hoch	^K	^K
A10	08	08	Cursor links	Cursor links	^H	^H
A11	09	09	Cursor rechts	Cursor rechts	^I	^I
F14	03	03	BREAK	BREAK	^C	^C
F02	1BH	1BH	ESCAPE	ESCAPE		
B13	0DH	0DH	ENTER	ENTER	^M	^M
F05	06	1EH	TAB-DUMMY	NL	^F	
D95	05	1CH	CRSR ANF.EDIT.	Fettdruck-DUMMY	^E	
B95	01	1DH	CRSR END.EDIT.	Norm.druck-DUMMY	^A	
E14	14H	13H	LIST (HC-BASIC)		^T	^S
D13	7FH	02	DEL-Editor			^B
F10	15H	0FH	RUN (HC-BASIC)		^U	^O
F11	19H	18H	CRSR Anf.-BASIC	CRSR End.-BASIC	^Y	^X
F12	12H	11H	INSERT HC-BASIC		^R	^Q
F13	10H	1FH	DEL HC-BASIC		^P	
E95	1CH	1CH	Fettdr.DUMMY	Fettdr.DUMMY		
F95	1DH	1DH	Norm.dr.DUMMY	Norm.dr.DUMMY		

Bild 4 Funktions- und Steuerzeichentasten
Die Steuerzeichen stehen mit in der Tastenbelegungstabelle und können somit anderen Erfordernissen angepaßt werden

Direktsprünge aus der Tastaturroutine heraus zu festen Adressen (100H, 200H, 300H), da dort die meisten Programme zu starten sind. Der Stack wird dazu vor der Ausführung des Sprungs neu initialisiert.

Eine weitere wichtige Taste ist die Screen-Copy-Taste (F09). Mit ihr ist es möglich, eine Bildschirmkopie auf dem Drucker zu erzeugen (Druckereinbindung über Sprungverteiler DRAKK), ohne das laufende Programm zu unterbrechen. Dabei ist zu beachten, daß während der Bildschirmkopie der Stack des laufenden Programms genutzt wird (Stackbelastung je nach Druckertreiber).

Die Tastenentprellung erfolgt positiv und negativ, das heißt, damit eine Taste als betätigt erkannt wird, muß mindestens 15 ms (2 MHz) lang ein prellfreies Signal anliegen (positive Entprellung). Das gleiche gilt übertragen für das Loslassen der Taste (negative Entprellung). Dadurch kommt es nicht vor, daß selbst bei sehr schlechten Kontakten eine Fehlfunktion auftritt.

Wird eine Taste länger als 0,5 s (2 MHz) betätigt, wird die Repeatfunktion (Echofunktion) bei Zeichentasten aktiviert. Der Normalaustritt aus der Tastaturroutine erfolgt kompatibel zur Originalroutine mit dem Zeichen im Akku und in Zelle 4. Eine „blinde" Tastaturabfrage erfolgt, indem vor dem Abruf der Routine die Zelle 4 auf 0 gesetzt wird.

Standard-Tastenbelegung

In den Bildern 4 bis 6 wird die Standard-Tastenbelegung gezeigt. Den Standort der Tastenbelegungstabelle kann man dem Tastenbelegungsfeldpointer (auf 3BH) entnehmen, der auf die Tastenbelegungstabelle im ROM zeigt (standardmäßig).

Aufbau des Pointerfeldes

Das Pointerfeld befindet sich auf dem Platz des jetzt nicht mehr benötigten Tastaturcodefeldes und wird vom Monitor initialisiert. Das originale Tastaturcodefeld beginnt auf Adresse 35H und wird durch den Monitorsprung RST 38H unterbrochen. Die ersten drei Zellen werden für Zählzellen des Druckertreibers genutzt und sind mit 0 initialisiert. Das Pointerfeld beginnt erst hinter dem RST 38H auf 3BH mit folgendem Aufbau:

- **3BH** – Tastenbelegungsfeldpointer
- **3DH** – Stringfeldpointer
- **3FH** – Pointer auf nächstes auszugebendes Stringzeichen
- **41H** – Länge Funktionstastenpositionsfeld
- **43H** – Funktionstastenpositionsfeldpointer
- **45H** – Funktionstastenadreßfeldpointer.

Aufbau des Stringfeldes

Das Stringfeld hat einen sehr einfachen und codeeffektiven Aufbau. Der auszugebende String beginnt mit dem Namen des Strings, welcher nur aus einem Grafikzeichen besteht. Es werden so lange Zeichen von diesem String ausgegeben, bis ein neues Grafikzeichen auftritt oder das Ende des Stringfeldes – gekennzeichnet mit einer 0 – erreicht ist. Damit ist auch klar, daß sich im String keine Grafikzeichen befinden dürfen, da diese für die Adressierung der Strings reserviert sind. Die Strings wie auch die Grafikzeichen können eine beliebige Länge haben und in beliebiger Reihenfolge angeordnet sein.

Stringfeldaufbau:

1. Grfk-Z. 1. String
2. erstes Stringz.

.
.
.

n. letztes Stringz.
n+1 Grfk. Z. 2. String
n+2 erstes Zeichen 2. String

.
.
.

0; Ende des Feldes

Günstig ist es, sich auf Kassette einige Felder vorzubereiten und diese mit einer kleinen Umladeroutine zu versehen, welche das Eintragen des Stringfeldpointers dem Standort entsprechend vornimmt. Somit ist das jeweilige Stringfeld nach einem Autostart sofort nutzbar.

Die für die Adressierung der Strings benötigten Grafikzeichen können in der Tastenbelegungstabelle stehen, welche bei Betätigung der entsprechenden Taste wirksam werden, worauf der dazugehörige String ausgegeben wird. Eine einfachere Möglichkeit besteht in der gleichzeitigen Betätigung von SHIFT und CTRL sowie einer Zeichentaste, worauf fol-

Bild 5 Tastenbelegungstabelle erste Shiftebene

Erste Shiftebene

Code	Pos.	Zeichen
31H	E1	1
51H	D1	Q
41H	C1	A
59H	B1	Y
32H	E2	2
57H	D2	W
53H	C2	S
58H	B2	X
33H	E3	3
45H	D3	E
44H	C3	D
43H	B3	C
34H	E4	4
52H	D4	R
46H	C4	F
56H	B4	V
35H	E5	5
54H	D5	T
47H	C5	G
42H	B5	B
36H	E6	6
5AH	D6	Z
48H	C6	H
4EH	A6	N
37H	E7	7
55H	D7	U
4AH	C7	J
4DH	B7	M
38H	E8	8
49H	D8	I
4BH	C8	K
2CH	B8	,
39H	E9	9
4FH	D9	O
4CH	C9	L
2EH	B9	.
30H	E10	0
50H	D10	P
5CH	C10	Ö
2DH	B10	-
7EH	E11	ß
5DH	D11	Ü
5BH	C11	Ä
3CH	E12	<
2BH	D12	+
23H	C12	#
5EH	E00	^
40H	B00	●
3EH	E13	>
0DH	B13	ENTER
0BH	A1	CURSOR HOCH
8	A10	CURSOR LINKS
20H	A5	SPACE
0	D0	SHIFT-LOCK ON/OFF
0AH	A0	CURSOR RUNTER
9H	A11	CURSOR RECHTS
0	B99,B11	SHIFT
0	C0	CTRL-TASTE
7FH	D13	DEL (ROTES C)
14H	E14	^T (ROTES L)
0	F01	MONITORRUECKKEHR
1BH	F02	ESCAPE
7BH	F03	{
7DH	F04	}
0	F06	JMP 100H
0	F07	JMP 200H
0	F08	JMP 300H
0	F09	CALL SCREEN-COPY
15H	F10	^U (RUN->BASIC)
19H	F11	^Y
12H	F12	^R (INSERT->BASIC)
10H	F13	^P (DEL->BASIC)
3	F14	BREAK=^C
6	F05	^F (TAB->EDITOR)
0	F53	GRAFIK ON/OFF
0	E53	HARDCOPY ON/OFF
0	D53	SLOW/FAST
0	C53	BEEP ON/OFF
1CH	E95	^\
1DH	F95	^]
5	D,C95	^E (WIPPE+)
1	B95	^A (WIPPE-)

Bild 6 Tastenbelegungstabelle zweite Shiftebene

Zweite Shiftebene

Code	Pos.	Zeichen
21H	E1	!
71H	D1	q
61H	C1	a
79H	B1	y
22H	E2	"
77H	D2	w
73H	C2	s
78H	B2	x
40H	E3	§
65H	D3	e
64H	C3	d
63H	B3	c
24H	E4	$
72H	D4	r
66H	C4	f
76H	B4	v
25H	E5	%
74H	D5	t
67H	C5	g
62H	B5	b
26H	E6	&
7AH	D6	z
68H	C6	h
6EH	B6	n
2FH	E7	/
75H	D7	u
6AH	C7	j
6DH	B7	m
28H	E8	(
69H	D8	i
6BH	C8	k
3BH	B8	;
29H	E9	)
6FH	D9	o
6CH	C9	l
3AH	B9	:
3DH	E10	=
70H	D10	p
7CH	C10	ö
5FH	B10	_
3FH	E11	?
7DH	D11	ü
7BH	C11	ä
5BH	E12	[
2AH	D12	*
27H	C12	'
7CH	E0	¦
5CH	B0	\
5DH	E13	]
0DH	C,B13	ENTER (CR)
0BH	A01	CURSOR HOCH
8	A10	CURSOR LINKS
20H	A05	SPACE
0	D00	SHIFT-LOCK ON/OFF
0AH	A00	CURSOR RUNTER
9	A11	CURSOR RUNTER
0	B99,B11	SHIFT
0	C00	CTRL-TASTE
2	D13	^B (ROTES C)
13H	E14	^S (ROTES L)
0	F01	MON. RUECKKEHR
1BH	F02	ESCAPE
60H	F03	`
7EH	F04	~
0	F06	JMP 100H
0	F07	JMP 200H
0	F08	JMP 300H
0	F09	BS-DRUCK
0FH	F10	^O
18H	F11	^X
11H	F12	^Q
1FH	F13	
3	F14	^C=BREAK
1EH	F05	NL (SIF1000)
0	F53	GRAFIK ON/OFF
0	E53	HARDCOPY ON/OFF
0	D53	SLOW/FAST
0	C53	BEEP ON/OFF
1CH	E95	
1DH	F95	
19H	D,C95	^Y (WIPPE+)
18H	B95	^X (WIPPE-)

gende interne Grafikzeichen erzeugt werden:

SHIFT-CTRL-Zeichentaste

A	81H	
B	82H	
C	83H	
D	84H	
E	85H	
F	88H	
G	87H	
H	8BH	
I	89H	
J	8AH	
K	9CH	
L	8CH	
M	8DH	
N	8BH	;DOPPELBEL. ZU H
O	8FH	
P	90H	
Q	91H	
R	92H	
S	86H	
T	94H	
U	95H	
V	88H	;DOPPELBEL. ZU F
W	97H	
X	86H	;DOPPELBEL. ZU S
Y	99H	
Z	9AH	
Ä	9BH	
Ö	9EH	
Ü	9DH	
0	90H	;DOPPELBEL. ZU P
1	91H	;-"- Q
2	92H	;-"- R
3	93H	
4	94H	;-"- T
5	95H	;-"- U
6	96H	
7	97H	;-"- W
8	98H	
9	99H	;-"- Y

Die Unregelmäßigkeiten zu den korrespondierenden CTRL-Zeichen entstehen beim gleichzeitigen Drücken von SHIFT und CTRL durch das Kurzschließen der Matrixzeile 7 und 8, da diese hardwaremäßig nicht entkoppelt sind.
Im EPROM wurde ein Standardstringfeld mit den am häufigsten vorkommenden BASIC-Schlüsselwörtern eingerichtet.

Direktzugriffe auf die Matrix

Obwohl das direkte Zugreifen auf die Tastaturmatrix von einem Anwenderprogramm aus unnötig und nur in sehr seltenen Fällen gerechtfertigt ist, gibt es eine Vielzahl von Programmen (meist Spielprogramme), die diese Methode anwenden. Um solche Programme ohne Änderung nutzen zu können, soll folgende Korrespondenztabelle dienen:

orig.	neu	Pos.
@	1	E01
A	3	E03
B	5	E05
C	7	E07
D	9	E09
E	B	E11
F	>	E13
G		
H	Q	D01
I	E	D03
J	T	D05
K	U	D07
L	O	D09
M	Ü	D11
N	ENT	C/B13
O		
P	Y	B01
Q	C	B03
R	B	B05
S	M	B07
T	·	B09
U	<	E12
V	←	A10
W		
S1	X	B02
S2	V	B04
S3	N	B06
S4	,	B08
←	-	B10
SPACE	@	B00
→	→	A11
ENT	CTRL	C00

Drei Tasten können nicht konvertiert werden, da diese Positionen bei der K 7659 nicht besetzt sind. Die Korrespondenztabelle kann auch dazu benutzt werden, um die Hardware mit dem originalen Tastaturprogramm zu testen.

Änderungen am Monitorprogramm

Eine der wichtigsten Änderungen ist das Eintragen des Aussprungs aus dem originalen Tastaturprogramm und der veränderten Initialisierung des ehemaligen Tastaturcodefeldes. Alles andere könnte original bleiben. Es werden aber weitere Änderungen vorgenommen, die nicht die Softwarekompatibilität verringern.
Solche Veränderungen sind:
– NMI-Unterbrechnung mit Registeranzeige (Init. 66H bei Reset)
–Hardcopyfunktion des Bildschirmtreibers
– Beschreiben eines I/O-Ports vom Monitor aus („0" ehemals „H")
– Auflistung der im Zusatzkommandoverteiler stehenden Kommandos („Z" ehemals „A")
– Initialisierung des Zusatzkommandoverteilers mit Standardwerten (@ L → Headersave load, @ S → Headersave save, @ D → Druckerreset, @ I → Druckerinitialisierung).

Da der zusätzliche Speicher mit dem Tastaturprogramm noch nicht ausgelastet ist, wurden auf dem Rest des Speichers das Headersave, das Screencopyprogramm, die Joystickabfrage sowie der Sprungverteiler für die Systemerweiterung fest installiert. Für die Übernahme der Software ist es günstig, die kompletten 4K ab F000H zu übernehmen.
Die umfassende Vorstellung des Sprungverteilers würde hier zu weit führen. Deshalb sollen nur die wichtigsten Sprünge beschrieben werden:
FFF7H – JMP STAT übergibt Tastaturstatus im Akku; A=0 – keine Taste gedrückt;
A=FFH – Taste gedrückt (außer SHIFT oder CTRL).
FFF4H – JMP SARUF ruft SAVE-Routine des Headersave; Parameterübergabe entsprechend Beschreibung Headersave.
FFF1H – JMP LORUF; Parameterübergabe entsprechend Beschreibung Headersave.
FFEBH – JMP DRDEL setzt den logischen Druckertreiber zurück (Zählzellen für Zeilen- und Spaltenposition werden zu 0).
FFE8H – JMP DRAKK übergibt den Akkuinhalt an den logischen Druckertreiber.
FFE5H – JMP BSDR druckt den Inhalt des BWS bis zur Cursorposition und kehrt in das aufrufende Programm zurück.
FFDFH – JMP DRZEL wie DRAKK, nur daß das Zeichen in Zelle 1BH übergeben wird (vorgesehen, um im BASIC mit POKE Zelle und CALL 0FFDFH zu drucken).
FFCDH – JMP DRINI Initialisierung des logischen Druckertreibers.
FFCAH – JMP ZEIDR übergibt ein Zeichen im Akku an physischen Druckertreiber.
FFBBH – JMP GETST Abfrage der Joysticks und Übergabe des Ergebnisses in BC (B-links, C-rechts) mit folgender Bit-Bedeutung (1):
Bit 0 – links
Bit 1 – rechts
Bit 2 – runter
Bit 3 – hoch
Bit 4 – Aktionstaste
Z-Flag=1, wenn keine Betätigung vorliegt
FFB8H – JMP SOUND; Ausgabe einer vollen Periode auf die Tonbandbuchse sowie auf Bit 7 vom Systemport;
Übergabe der Periodendauer in C mit $T = n*33\mu s - 20\mu s$ (2 MHz). Die Sprünge DRINI, DREZEL, DRAKK und ZEIDR sind nicht in den oberen 4K realisiert und mit JMP 0E800H, 0E803H, 0E806H und E809H initialisiert, wo man einen Unterverteiler für den verwendeten Druckertreiber installieren kann.

Vor- und Nachteile

Daß die beschriebene Tastaturvariante professionellen Ansprüchen gerecht wird, zeigen folgende Vorteile:
– geringer leistungsarmer Hardwareaufwand
– zwischen Tastatur und Rechner nur 10adriges Kabel erforderlich
– Anzeige der Tastaturmodes durch LEDs
– schaltbarer Tastenklick
– schnelle und sichere Abfrage
– Interruptfähigkeit
– nur noch zwei SHIFT-Ebenen
– hohe Softwarekompatibilität
– freie Belegbarkeit der Tasten
– frei programmierbare Stringausgabe mit Quittung
– Repeatfunktion auf allen Zeichentasten
– einfache Monitorrückkehr durch RST 38H–Taste
– direkter UP-Aufruf Screencopy aus der Tastaturroutine heraus
– Direktsprünge zu festen Adressen aus der Tastaturroutine heraus
– es können gegenüber der alten Routine alle Steuerzeichen erzeugt werden
– Abfrage der vollen 8∗12-Matrix
– einfache Erweiterung der Tastatur auf 96 Tasten, indem die nicht besetzten Positionen aufgefüllt werden.

Dem gegenüber stehen die Nachteile:
– Es wird zusätzlicher Adreßraum des Hauptspeichers benutzt.
– Es muß eine Änderung des originalen Monitorprogramms erfolgen.
– Nur bedingte Kompatibilität bei Direktzugriffen auf die Matrix und bei Sprüngen in die originale Routine hinein.
Betrachtet man Vor- und Nachteile, so kann festgestellt werden, daß es sich lohnt, die Nachteile in Kauf zu nehmen, zumal eine Änderung des Monitors nur einmal erfolgen muß. Die Tastatur ist also eine echte Alternative zu der Prozessortastatur des Herstellers, die einen höheren Hardwareaufwand erfordert und eine geringere Leistungsfähigkeit besitzt.

✉ KONTAKT ☏

Informatikzentrum des Hochschulwesens an der Technischen Universität Dresden, IG Heimcomputer der KDT, Mommsenstr. 13, Dresden, 8027; Tel. 4575240

Leiterplatte und Monitorprogramm plus Dokumentation können beim Computer-Club des VEB Robotron-Anlagenbau Leipzig, PSF 180, 7010 Leipzig (Tel. 7 16 15 78), zum Selbstkostenpreis bei Einsendung einer Tonbandkassette oder von EPROMs (4 KByte) sowie Rückporto bezogen werden. Der Computer-Club kann weiterhin die Software für die 8×8-Robotron-Tastatur K 7652 bereitstellen.

Schnelles Bildschirmlöschen beim KC 87

Schneller als der BASIC-Befehl CLS ist ein MC-Unterprogramm. In den Anwenderspeicher 220H bis 2FFH wird vor Beginn der Arbeit das Unterprogramm mittels eines kurzen BASIC-Programms geladen. Der Aufruf erfolgt mit CALL 544:

```
0220 LD HL, -5120
0223 LD M, 20H
0225 LD DE, -5119
0228 LD BC, 959
022B LDIR             ; BS löschen
022D LD DE, 0101H     ; 1. Zeile, 1. Sp.
0230 LD C, 18         ; UP Cursor
                      ; setzen
0232 CALL 5           ; BIOS-Ruf
0235 RET
```

```
10 FOR A=544 TO 565:READ
B:POKE A, B
:NEXT
20 DATA 33, 0, 236, 54, 32, 17, 1,
236, 1, 1
91, 3, 237, 176, 17, 1, 1, 14
30 DATA 18, 205, 5, 0,201
```

Das BASIC-Programm kann nach der Abarbeitung wieder gelöscht werden.

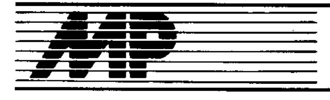
Peter Born

TERMINE

XV. Fachtagung des Fachausschusses 14 – Elektronische Steuerungen auf Mikrorechnerbasis

WER? Fachverband Elektrotechnik der KdT
WANN? 25. 10. 1988
WO? Berlin
WAS?
• Expertensystemmodi in der Projektierung moderner Steuerungssysteme
• 16-Bit-Industriecomputerfamilie ICA 700
• modulare Grundbaugruppen MGB 7000
• MRS 704/705; S 2000; P 8000
• Weiterentwicklung von audatec
WIE? Teilnahmemeldungen bitte an Kammer der Technik, Präsidium, Fachverband Elektrotechnik, Clara-Zetkin-Straße 115–117, Berlin, 1086

Hoppe



Entwicklungen und Tendenzen

Vertikalmagnetspeicher

Die magnetische Stabilität eines Einzelbits ist die entscheidende Größe für die Qualität eines magnetischen Speichermediums. Die Stabilität ist um so größer, je weiter Nord- und Südpol eines Magneten voneinander entfernt sind. Die Erhöhung der Datendichte führt bei konstanter Schichtdicke und gleichbleibender Spurbreite zu einer Abnahme des Längen-Dicken-Verhältnisses des magnetisierten Teilchens und somit verliert die gespeicherte Information an Stabilität. Konventionelle Medien weisen eine Grenze bei 500 Flußwechseln pro Milimeter auf.
Eine Erhöhung der Aufzeichnungsdichte bei Umgebung des reduzierten Längen-Dicken-Verhältnisses wird auf Medien der Vertikalaufzeichnung möglich. Bei diesen erfolgt die Magnetisierung senkrecht zur Schichtdicke.
Die Firma Toshiba entwickelte eine Diskette aus Barium-Ferrit. Das durch einen speziellen Glaskristallationsprozeß hergestellte Ba-Ferrit hat eine sehr hohe Koerzitivkraft (bis zu 6600 Oe). Deshalb muß es durch Dotierungen mit Co, Ni, Cu, Zn oder Ti gedrosselt werden, um für die Applikationen mit herkömmlichen Ferrit-Ring-Köpfen verwendet werden zu können. Die 3,5-Zoll-Diskette von Toshiba verfügt über 4 MByte Speicherkapazität. Die hohe Dichte der BaFe-Partikel bei durchschnittlicher Größe von 0,02 μm gestattet eine hohe Zuverlässigkeit aufgrund verminderter Oberflächenrauhigkeit. Die Massenproduktion soll auf bestehende Beschichtungsanlagen möglich sein, und es sollen keine Korrosionsprobleme auftreten.

Wi

aus Produktion 48/1987

Neues Material für löschbare optische Speicherung

Wissenschaftler des Phillips Forschungslaboratoriums in Eindhoven sollen eine vielversprechende Gruppe von Materialien für die löschbare optische Aufzeichnung sowohl analoger als auch digitaler Signale gefunden haben. Es sind Halbleitermaterialien wie Galliumantimonid (GaSb) und Indiumantimonid (InSb), die mit bestimmten anderen Elementen „dotiert" werden. Die spezifischen Eigenschaften ermöglichen es, Informationen mit einem Laserstrahl wiederholt aufzuzeichnen und zu löschen. Das Auslesen erfolgt wie bei den Informationsträgern Bildplatte und Compact Disc laseroptisch. Die Materialien können lange gelagert werden. Sie sind unempfindlich gegenüber normalen Umgebungstemperaturen und gegenüber Feuchtigkeit. Löschen und Aufzeichnen ist ungefähr tausend Mal möglich. Dies genügt privaten Anwendungen, ist aber für den professionellen Gebrauch unzureichend. Die Erforschung der Verfahren für die löschbare optische Speicherung ist nicht abgeschlossen. So werden als Ergebnis weiterer Untersuchungen Verbesserungen des Signal-Rausch-Abstandes erwartet.
Die Häufigkeit, mit der gelöscht und aufgezeichnet werden kann, muß erhöht werden. Zugleich wird nach weiteren Materialien aus der gleichen „Familie" gesucht, die eventuell noch bessere Eigenschaften haben. In diesem Zusammenhang könnten Tellur-Selen-Legierungen eine Rolle spielen. Diese könnten ebenfalls für die löschbare optische Speicherung nach dem Verfahren des kristallinamorphen Phasenübergangs geeignet sein, obwohl sie bisher für Anwendungen in der Praxis nicht schnell genug erneut kristallisieren.

Wi

aus Industrie-Elektrik und Elektronik 10/1987

Erste Testbauelemente auf Supraleiterbasis

Nachdem sich vor einem Jahr die Meldungen über die Entdeckung von Materialien mit Supraleiteigenschaften im Hochtemperaturbereich häuften, scheint nunmehr das Problem der Langzeitstabilität dieser Eigenschaft gelöst zu sein, so daß Testbauelemente gefertigt werden können.
So soll die amerikanische Firma TRW Inc. anstelle des Yttrium in dem bisher verwendeten Yttrium-Barium-Kupfer-Oxid das Element Erbium eingesetzt und dadurch eine stabile Supraleitung bei 92 K erreicht haben.
Die Firma AT&T Bell Laboratories soll ebenfalls einen Supraleiter gewonnen haben, aus dem sich Bauelemente herstellen lassen (77 K).
Durch General Motors Research Laboratories soll ein neues Verfahren zur Gewinnung supraleitender Dünnfilme entwickelt worden sein. Anstelle der Elektronenstrahlbedampfung erfolgt ein Abscheiden aus metallorganischen Verbindungen. Die Dünnfilme aus Yttrium-Barium-Kupfer-Oxid wurden auf einkristallinem Strontium und polykristallinem Bariumtitanat aufgebracht. Die Supraleitung soll bei 90 K auftreten.
Im GEC Forschungszentrum soll die Anwendung supraleitender Materialien als Leiterbahnen auf gedruckten Schaltungen vorgesehen sein.

aus electronics 61 (1988) 1 S. 32 und Elektronik.-München (1987) 26. – S. 7

Hierarchisch oder relational?

Die Anwendung der Datenbankmodelle wird sich von den hierarchischen zu den relationalen Modellen verlagern. Betrug im Jahre 1987 der Anteil der hierarchischen Modelle noch 78 Prozent, so sollen sich im Jahre 1992 folgende Anteile ergeben:

relationale Modelle	50 Prozent
hierarchische Modelle	35 Prozent
verteilte Modelle	15 Prozent.

Das relationale Datenbankmodell zeichnet sich dadurch aus, daß die Speicherung in Tabellenform flexibler an neue Anwendungsfälle angepaßt werden kann. Die Daten lassen sich schneller und einfacher definieren und verknüpfen. Redundanz sowie mangelnde Leistungsfähigkeit waren bisher Argumente gegen relationale Datenbankkonzepte. Die auftretende Redundanz hat sich in der Praxis nicht als problematisch erwiesen.
Für umfangreiche Datenbestände ist eine effektive Speicherung auch weiterhin mit hierarchischen Modellen vorteilhafter.
Verteilte Datenbankkonzepte haben das Forschungsstadium verlassen. Sie werden gegenwärtig in Pilotprojekten erprobt.

aus online.-Köln 33 (1988) 1. – S. 6

Erhöhung der Speicherkapazität

Obwohl Marktforschungsinstitute den 3,5-Zoll-Disketten-Speichern die größten Zuwachsraten voraussagen, werden weiterhin 5¼-Zoll- und 8-Zoll-Speicher angekündigt.
Die Firma Verbatim Corp. kündigte ein 12-MByte-Laufwerk mit 5¼-Zoll-Disketten an. Dieses Diskettenlaufwerk erreicht eine durchschnittliche Zugriffszeit von 65 ms und eine interne Transferrate von 2 MBit pro Sekunde. Als Diskettenmaterial verwendet die Firma ein Medium mit 600 Oersted, das eine Spurdichte von 333 Spuren/Zoll und eine Aufzeichnungsdichte von 19968 Bit/Zoll gestattet. Die verwendete Kunststoffhülle ermöglicht einen Betrieb im Temperaturbereich von 5 bis 70 Grad Celsius. Eine hohe Datensicherheit wird durch die Vorformatierung der Spurfolgeninformationen erreicht, die den Schreib-Lesekopf die exakte Spurlage überprüfen lassen.
Einen 8-Zoll-Festplattenspeicher mit einer Kapazität von 1000 MByte, einer durchschnittlichen Zugriffszeit von 16 ms, einer Transferrate von 3 MByte/s und einer MTBF von mehr als 35 000 Stunden stellte die Firma Fujitsu vor. Mit diesen Parametern zählt der Speicher zu den Spitzenprodukten. Im 5¼-Zoll-Bereich wurde von der Firma eine 390-MByte-Version mit 18 ms Zugriffszeit angekündigt. Das Angebot an 3,5-Zoll-Speichern wurde um ein Modell mit 77 Byte Speicherkapazität erweitert.

aus Computer-Zeitung. – Leinfelden-Echterdingen 19 (1987) 28. – S. 11

Konkurrenz zwischen Token-Ring-Architektur und Ethernet

Die physikalische Geschwindigkeit eines lokalen Netzes ist aus Anwendersicht nicht entscheidend für den Gebrauchswert. Als wesentlicher wird der Nettodurchsatz angesehen. Die Maximalgeschwindigkeit des Ethernet beträgt 10 MBaud, die des Token-Ring 4 MBaud. Der verwaltungstechnische Overhead der verwendeten Übertragungsprotokolle senkt die Transportleistung erheblich. Die Leistung des Ethernet wird dadurch von 10 MBaud auf 4 MBaud reduziert, während sie beim Token-Ring-Konzept nur auf 2,5 MBaud gebremst wird. Als leistungsmindernde Faktoren gehen auch die Adapterkarten und die Geschwindigkeit ein, mit der die Informationen vom Server geholt werden können.
Erst wenn die Geschwindigkeit unter 1 MBaud absinkt, kommt es zu einer Überlastung und zu Wartezeiten.
Unzureichend ist gegenwärtig auch die auf dem Markt befindliche Netzsoftware. Sie belastet die LANs mit unnötigen Anfragen und Redundanz oder die übertragenen Datenpakete sind zu klein, was den Verwaltungsaufwand vergrößert. Die leistungsstärkste Netzsoftware soll gegenwärtig zwischen 20 und 30 Prozent der verfügbaren Netzbreite ausnutzen.

aus PC-Woche v. 18. 1. 88, S. 15–16

Diode mit hoher Arbeitsfrequenz

Der bisherige Grenzwert der Resonanzfrequenz von Dioden soll durch Wissenschaftler der Universität von Illinois um mehr als das Dreifache übertroffen worden sein. Gegenwärtig sollen 100 GHz erreicht werden. Die Erfinder sehen eine Verbesserung bis in den Terahertzbereich für realisierbar an.
Die Diode soll aus Schichten von abwechselnd schwach dotiertem und stark dotiertem Aluminium-Gallium-Arsenid aufgebaut sein und wird mit Hochspannung betrieben. Dabei wird die Diode zur thermionischen Lichtemission gebracht und es kommt zu der sehr hohen Elektronenbeschleunigung. Die neue Diode soll in der Radartechnik zur Anwendung kommen.

Die Welt v. 16. 1. 88

Pläne für 64-Megabit-Chip

Anfang 1990 will Hitachi einen 64-MBit-DRAM unter Verwendung von 0,3 μm-Strukturen auf den Markt bringen. Die gegenwärtig von Hitachi produzierten 1-MBit-DRAMs werden mit konventioneller Fotolithografie in 1,5-μm-Strukturen erzeugt, die auch für 16-MBit-DRAMs eingesetzt werden soll. Beim 64-MBit-DRAM soll mittels Röntgenstrahllithographie eine dreidimensionale Struktur erzeugt werden.
Mitte der 90er Jahre soll der 64-MBit-Chip produktionsreif sein, den die Firmen Siemens AG und Philps gemeinsam im Rahmen des sogenannten Jessi-Projektes (Joint European Submicron Silicon Initiative) entwickeln wollen. Die Kosten dieses Projektes werden mit 3,5 Mrd. DM veranschlagt. Gegenwärtig arbeiten die Firmen an der Entwicklung eines 4-MBit-Chips.

aus Elektronic Weekly.-London (1987) 1393. – S. 48

Off-line Datentransfer zwischen A 7100 und A 7150

Es wurde eine Möglichkeit geschaffen, Disketten eines A 7100 (Quelldisketten) an einem PC 1715 einzulesen, die Daten dort zu transferieren und die somit erzeugten Zieldisketten an einem A 7150 einzulesen.

- Aktionen am PC 1715

– Initialisieren einer Ziel-Diskette unter CPA und FORMATP (Menüpfad: 0XYF) im „V..-Format" (708K).
– Transfer von POWER und FORMATP auf diese Diskette.
– Einstellen des Laufwerkes für die Quell-Diskette auf das SCP-Format.
– Transfer von der Quelle zum Ziel.

- Aktionen am A 7150

– Start von DCP mit einem aus der CONFIG.SYS-Datei heraus aufgerufenen Treiber.
– Einstellen des durch CONFIG.SYS bestimmten physischen Laufwerkes bzw. des durch die Gesamt-Konfiguration bestimmten logischen Laufwerkes ⟨Lw⟩ mit dem Kommando SELECT – I ⟨Lw⟩ cpc-2 auf das „V..-Format".
– Lesendes und schreibendes „Ansprechen" einer Diskette über den Buchstaben ⟨Lw⟩ mit beliebigen DCP-Kommandos.

Institut für Arzneimittelwesen der DDR, Dr. U. Pape, Große Seestr. 4, Berlin, 1120, Tel. 3651308

Eine MULTIDEZIMALSTELLEN-ARITHMETIK oder STRINGARITHMETIK für Kleincomputer

Für viele Probleme ist die Ziffernanzahl/Zahl im Kleincomputer zur Lösung nicht ausreichend. Für solche Aufgaben wurden auf STRING-Basis SUBROUTINEN geschrieben, die Addition (Subtraktion), Multiplikation, Division, Vorzeichenumkehr und absoluten Betrag realisieren. Die Operanden sind als STRING darzustellen in der Standardform: ⟨Vorzeichen⟩ ⟨ganzer Anteil⟩ ⟨Dezimalpunkt⟩ ⟨Bruchanteil⟩. Durch einen Parameter kann die maximale Länge vom Bruchanteil des Ergebnisses gesteuert werden. Damit können arithmetisch „DEZIMALZAHLENSTRINGs" verarbeitet werden, die aus maximal 253 Dezimalziffern bestehen können. Die Rechenzeit ist allerdings sehr hoch, zum Beispiel 5,5 Minuten für die Division von 1/3 auf 250 Dezimalstellen.
Gerätetechnik: KC 85/2 mit Modul 006 BASIC und Modul M022 EXPANDER RAM.
Programmumfang: insgesamt ca. 250 BASIC-Anweisungen.

Ernst-Moritz-Arndt-Universität Greifswald, Rechenzentrum, Friedrich-Ludwig-Jahn-Straße 15a, Greifswald, 2200

Dr. Dörband

Transformationsprogramm von dBase-II- in TURBO-PASCAL-Dateien

Das in MP 9/87 dargestellte Transformationsprogramm für dBase-II- und TURBO-PASCAL-Dateien hat unserer Meinung nach folgende Nachteile:
① Es lohnt sich aus Zeitgründen in den meisten Fällen nicht und führt leicht zu Fehlern, wenn man die transformierte und bearbeitete PASCAL-Datei wieder in eine dBase-II-Datei mit der gleichen Struktur zurücktransormiert.
② Die Dateigröße ist unnötigerweise eingeschränkt, weil sich sowohl die dBase-II-Datei als auch die transformierte PASCAL-Datei auf derselben Diskette befinden.
③ Die zwangsweise Umwandlung von numerischen dBase-II-Feldern der Länge 2 ohne Dezimalstellen in den PASCAL-Typ „Byte" führt zu Fehlern, falls die Feldinhalte negativ sind.
④ Das Löschkennzeichen der dBase-II-Dateien wird nur unzureichend berücksichtigt; es können dadurch Datenverfälschungen auftreten.
⑤ Es müssen alle Fehler der dBase-II-Dateien in die PASCAL-Datei übernommen werden.
Die von uns geschaffene Lösung der Transformation von dBase-II-Dateien in TURBO-PASCAL-Dateien mit dem Programm „dbpas" besitzt diese Nachteile nicht. Hier die wichtigsten Kenndaten:
– Nach dem Aufruf des Programms erfolgen alle Angaben menügesteuert.
– Die Laufwerke für Ziel- und Quelldatei sind beliebig.
– Es können alle Felder der dBase-II-Struktur oder nur ausgewählte Felder transformiert werden.
– Das Löschkennzeichen wird als Datenfeld behandelt.
– Schließt man das Löschkennzeichen von der Übertragung aus, ist die Übertragung der so gekennzeichneten Sätze wahlweise möglich.
– Die Umwandlung in die PASCAL-Satzfeldtypen ist zum Teil frei wählbar.
– Neben der Deklarationsquelldatei wird eine zweite Quelltextdatei angelegt, welche als Includefile von ersterer aufgerufen wird. So entsteht ein sofort lauffähiges PASCAL-Programm zum Auflisten der transformierten PASCAL-Datei. Darüber hinaus arbeiten wir an einer Programmlösung, um – ausgehend von der Deklarationsquelltextdatei oder einer vorhandenen dBase-II-Struktur – eine Rücktransformation von PASCAL-Dateien in dBase-II-Dateien zu ermöglichen.

VEB Fahrzeugausrüstung Berlin, Stammbetrieb des Kombinat Schienenfahrzeugbau, Abt. DPP, Adlergestell 598, Berlin, 1183; Tel. 6810377

Löblich

BC/PC – Lichtsatz – Druck

In einer Kooperationsvereinbarung zwischen dem KEAW Berlin und dem Berliner Verlag wurde festgelegt, für den Hauptteil der P8000-Dokumentation (WEGA-Programmierhandbuch, ca. 1300 Seiten), die Lichtsatzanlage DIGISET 400T30 zu nutzen.
Der mit dem Editorprogramm TP erfaßte Text befindet sich auf einer Minidiskette im SCPX-Format. Alle für die typographische Gestaltung des Textes notwendigen DOSY-Satzbefehle sind im Text enthalten.
Das Programm EDS (Erfassung für Datensichtgerät) übernimmt die Formatierung von Disketten in das DS2069-Format (für DIGISET 400T30) und die Konvertierung der Textdaten in den DS2069-Code. Außerdem können Texteinheiten der DS2069-Diskette angezeigt, in das SCPX-Format rückkonvertiert und auf Drucker ausgegeben werden.
Bei der Umwandlung von bereits erfaßten ASCII-Texten kommt man ohne eine abschließende manuelle Überarbeitung (Einfügen von DOSY-Satzbefehlen) und TP auf einem SCP-System oder am Datensichtgerät DS2069 nicht aus.
Dabei ist beim Programm EDS die Hervorhebung von DOSY-Satzbefehlen beim Auslisten der Datei über die Kommandos „L" (Bildschirm: inverse Darstellung) und „P" (Drucker: Doppel- oder Rotdruck) für den Nutzer besonders hilfreich.
Das Programm EDS schafft günstige Voraussetzungen zur Vergrößerung der Erfassungskapazität von Lichtsatzanlagen in den Satzzentren der Polygraphischen Industrie. Daneben wurde durch die Integration des Kommandos „K" die Möglichkeit geschaffen, bereits erfaßte Textdateien im ASCII-Kode in DS2069-Dateien umzusetzen.
Da bereits eine große Anzahl von Texten (Manuskripten u. a.) auf Personalcomputer existiert, besteht somit die Möglichkeit, aus diesen mit geringem zusätzlichen Aufwand qualitativ hochwertige Druckvorlagen herzustellen.
Der Weg, unter Nutzung eines SCP-Systems (z. B. PC 1715 oder A 7100) zu Lichtsatz-Druckvorlagen zu gelangen, kann für viele Anwendungsbereiche (von Vorlagen bis zu den Betrieben) eine kostengünstige Lösung darstellen.

VEB Datenverarbeitungszentrum Berlin, Storkower Straße 111, Berlin, 1055; Tel. 4351588

Dr. Schäfer

Generieren von sich selbst installierenden Zeichensätzen (SZS) für den KC 85/2/3

Der Programmkomplex dient dem Generieren und Testen neuer Zeichenbildtabellen auf dem KC 85/2/3. Dabei können jeweils die Zeichen der Codes 0–7F, 80–FF und 80–8B mit Funktionstastenbelegung generiert werden. Die Zeichenbildtabellen sind in einem SZS eingebunden und dieser wird auf Kassette gerettet. Der SZS kann in jeden KC 85/2/3 geladen werden und installiert den Zeichensatz ins Betriebssystem selbständig. Gleichzeitig sorgt er dafür, daß alle 256 Zeichen des Computers über die Tastatur erreichbar sind.
Der SZS kann für verschiedene Adressen generiert werden. Die Konstruktion der Zeichen erfolgt mittels Tastatur bei 8fach vergrößerter Zeichendarstellung.
Es existiert eine Generatorvariante, welche gleichzeitig mit dem BASIC-Interpreter verwendet werden kann.

Handelshochschule Leipzig, ORZ, Markgrafenstr. 2, Leipzig, 7010; Tel. 7481 App. 77

Zerbst

Kopplung MFA mit PC und Drucker über V.24

Das Modulare Fourieranalysatorsystem MFA (VEB Meßelektronik Berlin) stellt dem Anwender in der Verkaufsvariante die Schnittstellen CENTRONICS und SIF1000 zur Verfügung, wobei vom Betriebssystem die Arbeit mit Digitalkassetten- und Lochbandtechnik unterstützt wird. Es wurde eine Kopplung des MFA mit einem PC 1715 und einem Vollgrafikdrucker K 6314 über eine V.24-Schnittstelle realisiert. Im Vordergrund der Lösung stand ein minimaler Aufwand für die Hardware, für die ein empfangsseitig entsprechend aufgerüsteter KC 85/1-Druckermodul verwendet wurde. In beiden Geräten wurden Sende- und Empfangsroutinen installiert, wobei die bisher im MFA zur Verfügung stehende Software (Version 1.05 bzw. 1.07) ohne Einschränkungen nutzbar bleibt. Alle Routinen wurden so gestaltet, daß der Nutzer die zu übertragenden Speicherbereiche frei wählen kann. Zum Druck von alphanumerischen und grafischen Bildschirminhalten wurden im CENTRONICS-Treiber die hardwareseitigen Ausgaberoutinen verändert. Unter anderem wurde dabei die Interruptsteuerung durch Polling ersetzt. Die Lösung gestattet es, mit geringem Aufwand die Einsatzmöglichkeiten des MFA wesentlich zu erweitern: So können Meßwerte und Auswerteprogramme auf Diskette gespeichert sowie der PC als Wirtsrechner für MFA-Programmentwicklungen genutzt werden.

Offiziershochschule der Volksmarine „Karl Liebknecht" Stralsund, Lehrstuhl Informatik/Steuerungstechnik, PF 16119/G, Stralsund, 2300; Tel. 683372

Derlich/Havemann

Objektmodulbibliotheken für A 7100

Für die Arbeit mit FORTRAN 77 am A 7100 wurden Objektmodulbibliotheken für die Gestaltung des alphanumerischen Bildschirmdialogs und der grafischen Ausgabe erarbeitet.
Die Bibliothek SCREEN 1 gestattet dem Programmierer die Nutzung der Möglichkeiten des alphanumerischen Bildschirms, wobei besonderer Wert auf die Gestaltung der Ausgabe und die Erkennung von Eingabefehlern gelegt wurde, auf komfortable Weise.
Auf der Grundlage des ISO-Standards 7942 GKS wird in der Bibliothek GKSGX 1 eine Teilmenge des GKS-Niveaus 0a und 0b zur Verfügung gestellt. Die Bibliothek greift dabei

auf die Betriebssystemerweiterung SCP-GX für die vollgrafische Arbeit mit dem A 7100 zurück.

Technische Hochschule Leuna-Merseburg, Rechenzentrum/Bereich Informatik, Otto-Nuschke-Straße, Merseburg, 4200; Tel. 462922 (Dr. Reimann)

Prof. Dr. Weiß

MS-DOS-Dokumentation

Im Zusammenhang mit der Einsatzvorbereitung MS-DOS-fähiger Computertechnik wurde eine umfangreiche Dokumentation erarbeitet, die maschinenlesbar nachgenutzt werden kann. In der Dokumentation werden schwerpunktmäßig der Kommandovorrat, die Arbeit mit Editoren und dem Linker, Konfigurations- und Stapelverarbeitungsdateien, Dienstprogramme und die Technologie der Assemblerprogrammierung behandelt.

Technische Hochschule Leuna-Merseburg, Rechenzentrum/Bereich Informatik, Otto-Nuschke-Straße, Merseburg, 4200; Tel. 462922 (Dr. Reimann)

Prof. Dr. Weiß

SCPTEXT 1700

Das Programm SCPTEXT 1700 dient der Eintragung einer Autostartroutine in das Betriebssystem SCP 1700. Für sogenannte Systemdisketten ist es oft wünschenswert, wenn nach Einschalten des Rechners, eines Systemresets oder nach Ferneinschalten ein bestimmtes Programm sofort gestartet wird, wie z. B. das residente Kommando DIR oder Diskettenfiles wie DBASE, TABCALC o. a.
Das Programm SCPTEXT 1700 realisiert das Eintragen des gewünschten Befehls in den im Betriebssystem SCP 1700 reservierten Puffer ohne Zuhilfenahme eines Debuggers (Rechner A 71**). Je nach Aufruf des Programmes kann der Starttext im Dialog oder direkt von der Kommandozeile eingegeben werden.
Ein Kopieren von einem Laufwerk auf ein anderes bzw. mit gewechselter Diskette während des Eintragens ist möglich. Zum Lieferumfang gehören zwei modifizierte Betriebssysteme, bei denen die Dialogabfrage nach dem Laufwerk E übergangen wird und dieses selbständig initialisiert bzw. nicht eingerichtet wird.

Staatliche Hauptgasverteilung beim Ministerium für Kohle und Energie, Beilsteiner Str. 115, Berlin, 1140; Tel. 5473041 (Koll. Hartwig)

Pflieger

Listendruck- und Anzeigeprogramm

VALI ist ein variables Listendruck- und Anzeigeprogramm für ausgewählte Felder beliebiger TURBO-PASCAL Dateien in Tabellenform.
Das Programm ist anwendbar sowohl für Dateien, die mit TACCESS.BOX erstellt wurden als auch für normale TURBO-PASCAL Dateien. Die Felder können folgende Formate haben: STRING, INTEGER, REAL, BOOLEAN, CHAR und ARRAY[1...] OF REAL. Die für den direkten Zugriff so wichtige INTEGER-Darstellung im STRING-Format ist ebenfalls möglich. Die maximale Anzahl der Felder der Datei ist auf 256 begrenzt worden (einschließlich der Vektorfelder). Angezeigt werden außerdem die Satzlänge und die physische Satzanzahl der Datei. Das Prinzip beruht darauf, daß zu jeder Datei eine Textdatei mit der Beschreibung des RECORDS angelegt wird, bzw. es müßte eine solche Datei schon beim Programmierer existieren. Die Ausgabe ist wahlweise auf Drucker oder Bildschirm möglich. Will man sich einzelne Felder von Vektoren anzeigen lassen, erfolgt es in der Form: best[3] oder best[11], wie sie in TURBO-PASCAL üblich ist.
Die Feldnamen werden als Tabellenüberschrift mit angezeigt, außerdem kann eine Überschriftzeile eingegeben werden. Das Programm läuft unter SCP auf PC 1715, BC A 5120/30 und AC A 7100. Es ist geplant, dieses Programm noch weiterzuentwickeln, u. a. die arithmetische Verknüpfung von Feldern (z. B. FELD a + FELD b oder FELD c * FELD d). Nachnutzer dieses Programms erhalten dann im Nachgang jeweils die neueste Version.
Bei Zusendung einer Diskette (8" oder 5 1/4") liefern wir das Programm VALI.COM und die Dokumentation VALI.DOC.

DEWAG Leipzig, Abt. DO, Oststrasse 105 Leipzig, 7050

Mikrorechnerüberwachung mit softwaregeschütztem Anlauf

In allen Mikrorechnern, besonders in der Industrieautomation und bei Anwendung von Watchdogschaltungen, ist eine geschützte ununterbrochene Initialisierung im Rechneranlauf sehr wichtig.
Ohne zusätzliche Schaltungen und ohne Abschaltung von Rechnersignalen garantiert der softwaregeschützte Anlauf die sichere Initialisierung des Mikrorechners. Durch zusätzliche Steuerwörter wurde ein System entwickelt, welches zu jedem Zeitpunkt nachteillos unterbrochen werden kann. Der softwaregeschützte Anlauf kann in allen Mikrorechnern ohne Geräteveränderung implementiert werden. Diese Maßnahme wurde im Kleincomputer PCU 01 für Prozeßdatenerfassung und -überwachung unter harten Industriebedingungen umfassend erprobt.

VEB Rationalisierungsmittelbau Wolle und Seide Greiz, Bereich F. u. E, Zeulenrodaer Str. 13–15, Greiz, 6600, Tel. 3155 (Koll. Stiebert)

Fritzsche

Programmierbarer Zeichengenerator für den KC 85/1 und KC 87

Die Hardwarekonfiguration des Kleincomputers KC 85/1 ermöglicht bisher eine beschränkte, nicht frei programmierbare, quasigrafische Darstellung der auf einem EPROM vorhandenen Grafikzeichen (ASCII-Code 128 bis 255 Dez.).
Mit der Hardware-Ergänzung „Programmierbarer Zeichengenerator" (PZG) können durch den Nutzer 126 Grafikzeichen frei programmiert (Erzeugung der gewünschten Bitmuster), auf dem Bildschirm dargestellt und ggf. über einen Drucker (K 6313 o. ä.) ausgegeben werden.
Der PZG besteht aus einer kleinen Leiterplatte mit einem zusätzlichen 1-KByte-RAM (2×U 214), einem herkömmlichen Zeichensatz auf einem 2-KByte-EPROM (224 programmierte ASCII-Zeichen 32 bis 255 Dez.) und einem Flip-Flop (DL 074) zur wechselseitigen Umschaltung zwischen der Standard-Grafik und der frei programmierbaren Grafik.
Die Ansteuerung des PZG erfolgt über 3 Schaltadressen, die sich im normalerweisen nicht zugänglichen Farb-RAM als Kurzmerkspeicher befinden. Sie dienen zum Einschalten des PZG, zum Beschreiben des PZG-RAM und zum Aktivieren des PZG-RAM.
Die realisierte Installierung der PZG-Leiterkarte, die zweckmäßigerweise an Stelle des Farbmoduls (der Farbmodul befindet sich auf der PZG-Karte) unter Nutzung der Standard-Steckverbindung vorgenommen wurde, verändert äußerlich den KC 85/1 nicht und sichert die herkömmliche Funktionsfähigkeit des Kleincomputers.
Es bestehen die Möglichkeiten der Verwendung verschiedenartiger Zeichensätze, z. B.:
- grafische Funktionsdarstellungen
- kyrillische Buchstaben (Russisch-Zeichensatz)
- lateinische Schreibschrift und
- Darstellung elektronischer Schaltungen.

Ergänzende Software ermöglicht die selbständige Programmierung von Grafikzeichen (MC-Programme) und das Ausdrucken der erzeugten Bildschirminhalte.
Die Betreibung des PZG ist sinnvollerweise mit Farbmodul (aber auch ohne Farbfernsehgerät und RGB-Satz) zu realisieren.
Neben einer selbstgefertigten PZG-Leiterkarte liegen eine Kurzdokumentation zur Hardwarelösung und Programmbeschreibung beispielhafter Demonstrations-Software vor.

Technische Universität Magdeburg, Büro für Neuererwesen (oder Sektion Apparate- und Anlagenbau, WB 2) PF 14, Magdeburg, 3010; Tel. 5920

Bundrock

REDABAS-Transferprogramme

Zur Nachnutzung bieten wir für die Mikrorechnertypen PC 1715, BC A 5120 und AC 7100 unter dem Betriebssystem SCP und Kompatiblen das Transferpaket REKP an.
Eine REDABAS-Datei wird in eine Rettedatei für die Kalkulationsprogramme KP, SC bzw. TABCALC/M 16 umgewandelt und kann durch diese geladen werden.
Beliebige Felder der REDABAS-Datei (auch Teilketten der Felder) können in beliebige Zeilen und Spalten der Tabelle transferiert werden.
Im Anschluß an die REDABAS-Daten können Werte (z. B. für eine Summenzeile) vordefiniert und in die Tabelle mit eingetragen werden.
Das Programmpaket besteht aus einem Vorprogramm (REKP1), mit dessen Hilfe der Anwender im Dialog den Transfer beschreibt. Die Beschreibung wird in einer Textdatei abgelegt und von dem Nachfolgeprogramm (REKP2) zum Entladen der REDABAS-Datei benutzt. Wiederkehrende Transferaufgaben können auf diese vorgefertigten Beschreibungsdaten zurückgreifen.
Die Beschreibung einer Transferaufgabe dauert etwa 5 bis 10 Minuten und der Entladevorgang rund 1 Minute. Der Ladevorgang im Kalkulationsprogramm kann in Abhängigkeit vom Umfang der zu ladenden Daten mit 1 bis 4 Minuten (mehr als 5000 Elemente bei TABCALC/M 16) angenommen werden.
Zur Unterstützung des Rücktransportes vom Kalkulationsprogramm zu REDABAS kann das Selektions- und Formatierprogramm SELEKTOR mitgeliefert werden. Das in eine sequentielle Textdatei entladene Tableau kann mit Hilfe von SELEKTOR an SDF-Formate beliebiger REDABAS-Dateien in einem Arbeitsgang angepaßt werden. Die Selektion bestimmter Zeilen anhand vorgegebener Wertebereiche für beliebige Spalten ist möglich. Die Parameter von SELEKTOR werden in einer Textdatei programmiert, so daß der Vorgang automatisiert ablaufen kann.

VEB Landmaschinenbau Güstrow, Abt. L102, Rövertannen, Güstrow, 2600; Tel. 46515 (Koll. Ansorge)

Dr. Falk

Wir suchen ...

... eine Zeilenbefehlserweiterung des BASIC-Interpreters analog der vorgestellten Lösung für KC in MP 11/87 für unsere PC 1715.

VEB Bandstahlkombinat „Hermann Matern", Eisenhüttenkombinat Ost, Zweigbetrieb Bandstahlveredlung Porschdorf, Porschdorf, 8321, Tel. Bad Schandau 2556/2558, App. 75

Polei

... einen Drucker-Driver zum Anschluß eines Commodore-Druckers an U-880-Rechner (MC 80.20).

VEB Keramische Werke Haldensleben, HA Rationalisierung, Karl-Marx-Straße, Haldensleben, 3240; Tel. 56350 (Koll. Trebesius)

Richter

... eine Software-Lösung zur Kopplung von IBM-XT-kompatibeln PC und KBR K1630 über MUX20.

VEB MAW Armaturenwerk Prenzlau, Abt. DF. Wilhelm-Pieck-Str. 91, Prenzlau, 2130; Tel. 32218

Müller

... Informationen bzw. Hinweise zum Betreiben eines Zeilendruckers VT 27000 am BC A 5120. Großhandelsbetrieb Waren täglicher Bedarf,

Fachdirektion Ökonomie, Pausaer Straße 15, Plauen, 9900; Tel. 27022, App. 29

... eine Kopplung PC 1715–KC 85/3. An unserer Schule ist ein Computerkabinett mit 19 KC 85/3 und einem PC 1715 eingerichtet worden. Die KC und ein Großbildschirm (Farbfernsehgerät mit FBAS- und RGB-Eingang) sind über ein Dialogsystem miteinander gekoppelt. Um den Einsatz des PC zu demonstrieren, würden wir gern die Verbindung PC/KC (ohne V.24-Schnittstelle) realisieren.
Wer kann helfen?

Kommunale Berufsschule „Martha Brautzsch" Merseburg, 4200 Merseburg, Oberaltenburg 4

Schürhold



Literatur

Programmieren mit C

von M. Clauß und G. Fischer
Reihe Technische Informatik. VEB Verlag Technik Berlin, 1988, 240 Seiten, 14 Bilder, Broschur, 24,– M

Mit dem nunmehr vierten Band der Reihe „Technische Informatik" wird vom VEB Verlag Technik einem breiten Bedürfnis nach einer umfassenden und methodischen Darstellung der Programmiersprache C Rechnung getragen.
Die Darstellung von C erfolgt in Form eines Lehr- und Arbeitstextes, der das Erlernen der Programmiersprache im Selbststudium ermöglicht. Dies wird durch eine Reihe von Übungsaufgaben, die zur praktischen Arbeit mit C anregen, unterstrichen. Es fehlt aber eine Besprechung der gestellten Aufgaben.
Im ersten Kapitel werden grundsätzliche Aspekte der Sprache C behandelt, so daß schnell die Voraussetzungen für praktische Programmierübungen gegeben sind. Im zweiten Kapitel folgt die ausführliche Darlegung der Sprachkonzepte von C, die durch eine Vielzahl von Beispielprogrammen effektiv unterstützt wird. Im abschließenden dritten Kapitel wird auf verschiedene Aspekte der Anwendung von C, z. B.: C-Präprozessor, Arbeit mit der Standardbibliothek, Beispiele für systemnahe Programmierung eingegangen.
Ein umfangreicher Anhang (54 Seiten) enthält eine Zusammenfassung der Syntax von C, eine Beschreibung des durch X/OPEN definierten Umfangs der C-Standardbibliothek sowie eine kurze Erläuterung des unter UNIX verfügbaren Prüfprogramms „lint". Die Autoren berücksichtigen auch internationale Standardisierungsbemühungen (ANSI-C und X/OPEN-C) und beschreiben Erweiterungen und Änderungen gegenüber K&R.
Den Autoren ist es gelungen, sowohl für den Einsteiger in C als auch für den täglichen Nutzer ein wertvolles Arbeitsmittel geschaffen zu haben. Dem Verlag Technik ist zu danken, daß er auf aktuelle Trends der Softwareentwicklung mit diesem Band so prompt reagiert hat.

Ulrich Oefler

Hilfsmittel für Errichtung und Betrieb von Mikrorechner-Automatisierungsanlagen

Peter Neumann, Reihe Automatisierungstechnik: Band 219, VEB Verlag Technik Berlin, 1986, 88 Seiten, 4,80 M

Dieser Band verbirgt hinter dem einfachen Begriff ‚Hilfsmittel' eine gelungene Darstellung von ausgewählten wissenschaftlichen Verfahren und technischen Mitteln für die Vorbereitung und Realisierung von mittleren und großen MR-Automatisierungsanlagen. Als Leser werden Automatisierungstechniker und Informatiker angesprochen, die sich sowohl mit der Entwicklung als auch mit der Einsatzvorbereitung, insbesondere Projektierung, aber auch mit dem Betrieb solcher Anlagen befassen. Zur Einordnung seines Anliegens geht der Autor von seinen praktischen Erfahrungen im Automatisierungsanlagenbau aus und charakterisiert die hauptsächlichen Arbeitsprozesse. Behandelt werden die Erarbeitung von Grund- und Automatisierungskonzeption, Ausführungsprojektierung, Arbeitsschritte des Herstellungsprozesses sowie einige Aufgaben zu Betrieb und Instandhaltung.
Darauf aufbauend werden vorzugsweise für die technische Vorbereitung und Realisierung verschiedenartige Verfahren und Mittel (organisatorische, wissenschaftliche und technische) beschrieben. Ausführlicher wird ein Verfahren behandelt zur Echtzeitverhaltensanalyse von MR-Einrichtungen und Datenübertragungssystemen auf Grundlage von Bedienungsmodellen. Verfahren und Mittel zur Zuverlässigkeitsanalyse sowie Maßnahmen zur Synthese fehlertoleranter MR-Automatisierungsanlagen vermitteln einen guten Einblick in die Problematik.
Mit diesem Band wird der interessierte Leser sehr effektiv in wesentliche Problemkreise großer MR-Automatisierungsanlagen eingeführt. Dabei werden vorzugsweise Lösungswege dargestellt, die aus wissenschaftlichen Arbeiten des Autors hervorgegangen sind.

Dr. Ulrich Engmann

Von der einfachen Logikschaltung zum Mikrorechner

von Joachim Matschke, 3., bearbeitete Auflage, VEB Verlag Technik Berlin, 1986, 232 Seiten, 20,– M

Das Buch wird vor allem Zuspruch bei all denen finden, die sich möglichst einfach und schnell in die Grundlagen digitaler Mikroelektronik, vor allem moderner Mikrorechner einarbeiten wollen – oder gar müssen.
Für die Entwicklung und funktionelle Gestaltung elektronischer Geräte sowie für die Entwicklung und Einführung automatisierter Prozesse mit Mikroprozessoren und Mikrorechnern müssen heute verschiedene Spezialisten herangezogen werden. Jeder benötigt die Mitarbeit der anderen. Dabei wird es volkswirtschaftlich immer notwendiger, zur effektiven Nutzung der neuen Technik Einblicke in die Funktionsweise des Mikroprozessors und Mikrorechners in größter Breite und in verstärktem Maße zu vermitteln.
Um aber möglichst viele ansprechen zu können, dürfen nur geringe Kenntnisse vorausgesetzt werden und ist das nötige Wissen möglichst einfach und didaktisch geschickt aufzubereiten.
Diesem großen Ziel wird das vorliegende Buch sehr sorgfältig und effektiv gerecht. Das Buch füllt die vorhandene Lücke zwischen Publikationen populärwissenschaftlichen, überblickverschaffenden Charakters und fachwissenschaftlichen, meist speziellen Veröffentlichungen.
Mit bewußt so klein wie möglich gehaltenem mathematischem Aufwand und ebenso bewußtem Übergehen auch oftmals wichtiger Einzelheiten erleichtert das Buch ein Eindringen in die Problematik.
Diese gewollte Selbstbeschränkung auf wesentliche Sachbezüge ist beispielhaft und in ihrer konsequenten Durchsetzung ein geschickt eingesetztes Mittel, effektives Einarbeiten in die Thematik zu unterstützen.
Der angesprochene Leserkreis wird das Buch zu Studienzwecken, insbesondere auch durch die das Wesentliche herausstellende typografische Gestaltung, mit Gewinn nutzen.
Der stürmischen Entwicklung der Mikroelektronik Rechnung tragend, liegt das Buch in dritter, überarbeiteter und um Wirkprinzipien neuer Generationen von Mikroprozessoren ergänzter Auflage vor.

Dr. Ingo Schreiber

Programmieren in PL/1

von Henning Schoch, Band 8 der Reihe Informatik – Kybernetik – Rechentechnik, Akademie-Verlag Berlin 1987, 2., berichtigte Auflage, 521 Seiten

Wenn auch heute, im Gegensatz zur Zeit vor etwa 5 bis 10 Jahren und mitbedingt durch den großen Aufschwung, den die Kleinrechentechnik erlebt, kaum mehr ein Absolvent einer Hoch- oder Fachschule während seines Studiums nennenswerte Informationen über die Programmiersprache PL1 erhalten hat, hat die Überarbeitung des Buches von Schoch und seine Neuherausgabe dennoch – oder gerade – seine Berechtigung. War doch diese Sprache insbesondere dazu geeignet, die verschiedenen bis dahin entstandenen Sprachen für technische und ökonomische Aufgabenstellungen zu einer Einheit zusammenzufassen. Die Tendenz, daß der Ingenieur in wachsendem Maße neben der Behandlung wissenschaftlich-technischer Aufgabenstellungen mit Fragen der Technologie und den auch dort zu bewältigenden großen Datenmengen befaßt wird, daß der Ökonom in zunehmendem Maße mathematische Methoden in die Behandlung seiner Prozesse einbeziehen muß, machten eine solche Sprache wünschenswert, die darüber hinaus auf die hardwareseitigen Gegebenheiten der ESER-Großrechnertechnik zugeschnitten ist. So hatte diese Sprache auch bald in diese Bereiche Eingang gefunden und wird auch noch eine Zeitlang dort ihre Bedeutung behalten. Es ist deshalb für alle dort Tätigen angenehm, über ein Buch zu verfügen, das von einem mit der Sprache Erfahrenen geschrieben wurde und in das er sich bei vorhandenen Grundkenntnissen über problemorientierte Programmiersprachen selbständig einarbeiten, das er als Begleitbuch für spezielle Programmierlehrgänge oder auch als Nachschlagewerk benutzen kann.
Es ist besonders zu begrüßen, daß die Neuauflage im Akademie-Verlag (die erste Auflage mit dem Titel „Programmierung in PL/1" umfaßte nur die Teilsprache Subset-PL/1 und war beim Teubner-Verlag Leipzig erschienen) die komplette Sprache darbietet. Diese Veränderung, bei der man die 15 Jahre zusätzlich gewonnener Erfahrung wohltuend spürt, kann als gut gelungen bezeichnet werden. Das Buch hat eine klare Gliederung und übersichtliche Darstellung des Textes, so daß man es gern liest. Das gut gegliederte Inhaltsverzeichnis und das ausführliche Register erleichtern das Auffinden spezieller Einzelheiten beträchtlich. 175 Übungsaufgaben, gegenüber der 1. Auflage erweitert und aktualisiert (mit Lösungen), erleichtern ein selbständiges Durcharbeiten des Buches. Aber auch der Kenner erhält durch die Aufgaben manche Anregung. In den wenigen die Programmierung unterstützenden Skizzen, insbesondere in dem Abschnitt über die Steuerung des Programmablaufs, wäre bei der Behandlung der Algorithmusgrundstrukturen die Verwendung von Struktogrammen anzuraten gewesen, zumal die Sprache PL1 in umfassender Weise eine strukturierte Programmierung zuläßt. Es hätten sich hierdurch manche Vorteile im Hinblick auf Gliederung, Testung und Änderung von Programmen noch klarer herausarbeiten lassen.
Begrüßenswert ist, daß nicht nur die Sprache PL1 vermittelt wird, sondern auch die erforderlichen Beziehungen zum Betriebssystem dargestellt sind, um ein Programm auf einer Rechenanlage zum Laufen zu bringen. Leider geschieht das – bei allem Verständnis, daß hier Einschränkung geboten ist – nur für den Stapelbetrieb. Die wachsende Bedeutung, die hier auch der Dialog mit dem Großrechner über Terminals erlangt, findet bedauerlicherweise keine Berücksichtigung.
Alles in allem ist ein Buch entstanden, das gegenüber der 1. Auflage erheblich gewonnen hat und das dem in der einschlägigen Praxis Tätigen nützlich werden wird.

Prof. Dr. Günter Bräuning

TERMINE

6. Fachtagung Standardisierung

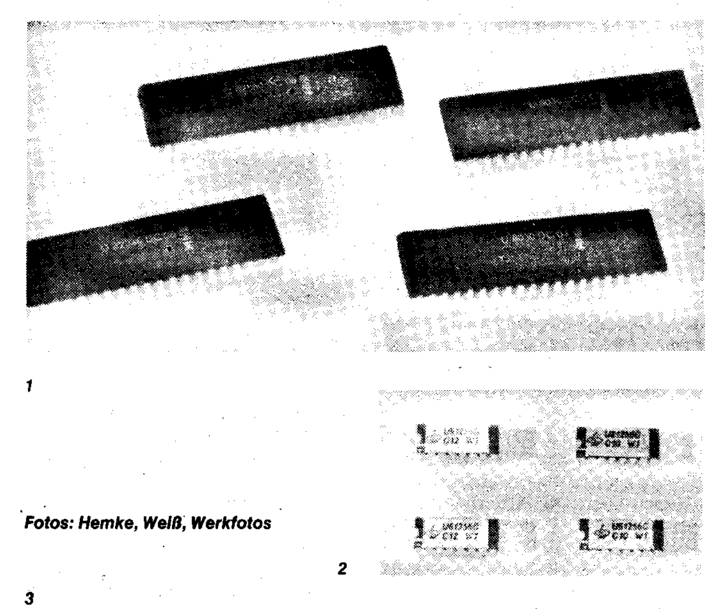
WER? Fachverband Elektrotechnik in der Kammer der Technik, Kommission Standardisierung
WANN? 10. bis –11. November 1988
WO? Leipzig, Brühlzentrum
WAS?
– CAD, CAM, CAQ
– Probleme der Softwarestandardisierung in der DDR
– Lichtwellenleiter-Nachrichtenübertragung
– Mitarbeit der DDR im RGW, in der IEC, der ISO, internationalen Organisationen und Zertifizierungsorganen
– Informationen über erworbene Zertifikate mittels Messen, Ausstellungen, Kataloge, Verpackungen, Inserate, Betriebsanleitungen
WIE? Vorträge bis zum 15. Juni 1988 sowie Teilnahmemeldungen sind schriftlich an folgende Anschrift zu richten: Kammer der Technik, Präsidium, Fachverband Elektrotechnik, Clara-Zetkin-Str. 115/117, Berlin, 1086, Tel. 2202531, App. 216

Hoppe



Leipziger Frühjahrsmesse 1988

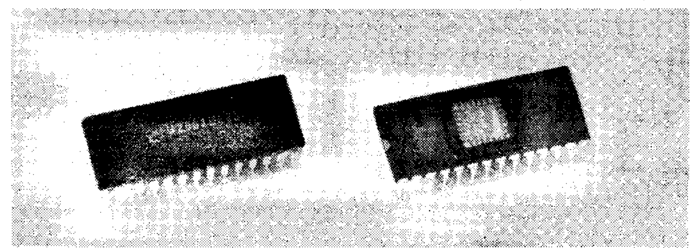
Ein hochqualifiziertes Angebot von Erzeugnissen aus 9 Branchenkomplexen der Investitionsgüter und 7 für Konsumgüter zeigten zur diesjährigen LFM rund 9000 Aussteller aller Kontinente. Im Blickpunkt des Technikangebotes stand das Thema „Integrierte Meß-, Prüf- und Regeltechnik – Weg zur Leistungssteigerung", dem vorrangig auch das wissenschaftlich-technische Fachvortragsprogramm und das Internationale Messesymposium gewidmet war. In der auch für dieses Gebiet wichtigen Branche Informations- und Kommunikationstechnik hatten Hauptanteil am DDR-Ausstellungsprogramm die Kombinate Robotron, Mikroelektronik und Carl Zeiss JENA.
In diesem ersten Teil des Messeberichtes werden wir versuchen, einen Überblick über die wesentlichen Bauelemente der Computertechnik und die ausgestellten Computer zu geben; der zweite Teil in Heft 8/88 beinhaltet dann verschiedene Applikationen, lokale Netze und periphere Geräte.



1

2

Fotos: Hemke, Weiß, Werkfotos



3

Bauelemente

- **Mikroprozessorschaltkreise**

Das Kombinat Mikroelektronik stellte für die 16-Bit-Systeme U 8000 und K 1810 WM86 die seriellen Interfacebausteine SCC **U 8030 DC 04** bzw. **U 82530 DC 06** sowie die kombinierten Bausteine mit Zähler/Zeitgeber und parallelem E/A-Port CIO **U 8036 DC 04** bzw. **U 82536 DC 06** im 40poligen Dual-in-Line-Duroplastgehäuse vor (Bild 1). Da die Schaltkreise U 82530 und U 82536 über einen nicht gemultiplexten Datenbus verfügen, sind sie auch für den Einsatz in den Mikroprozessorsystemen U 880 und I 8080 geeignet. Der SCC (Serial Communications Controller) besitzt zwei unabhängige serielle Vollduplex-Kanäle mit einer Übertragungsgeschwindigkeit von 0 bis 1 MBaud und mit zwei separaten Taktgeneratoren. Die CIO (Counter/Timer and Parallel I/O Unit) verfügt über zwei unabhängige, zweifach gepufferte, bidirektionale 8-Bit-Ports und ein 4-Bit-Spezialport. Programmierbar sind hierbei u. a. die Polarität, die Richtung (Bitmode) und die Ausgänge als open drain. Zwischen 4 Handshake-Modes, u. a. 3-Wire-Mode (für IEC-625-Bus), kann gewählt werden. Weiterhin beinhaltet die CIO drei unabhängige 16-Bit-Zähler/Zeitgeber mit programmierbarer Ausgangssignalform und Retriggerbarkeit. Die Schaltkreise U 8030 und U 8036 sind zum Z 8030 bzw. Z 8036 kompatibel, während die Kompatibilität des Schaltkreises U 82530 zum I 82530 sowie des Schaltkreises U 82536 zum I 82536 gegeben ist.
Für die Ansteuerung von maximal 4 Floppy-Disk-Laufwerken mit 8"- bzw. 5 1/4"-Disketten dienen die Floppy-Disk-Controller **U 8272 D 08** (8-MHz-Takt) bzw. **U 8272 D 04** (4-MHz-Takt). Zu diesen für die Mikroprozessorsysteme U 880, K 1810 WM86 und I 8080 A geeigneten Controllern finden Sie weitere Angaben in MP 4/88 sowie in diesem Heft.
Neu vom Kombinat Mikroelektronik waren weiterhin die 8-Bit-A/D-Wandler **C 670 C** und **C 670 Cn**. Sie arbeiten nach dem Prinzip der sukzessiven Approximation mit einer Umsetzzeit von 25 µs und einer Genauigkeit von 1 LSB bzw. 1/2 LSB. Diese A/D-Wandler-Schaltkreise beinhalten eine Referenzspannungsquelle, einen Taktgenerator sowie einen Enableeingang für die Tri-state-Ausgänge. Folgende D/A- bzw. A/D-Wandler wurden erstmals in einem Keramikgehäuse vorgestellt:

C 560 C (D/A; 8 Bit)
C 565 C (D/A; 12 Bit)
C 5650 C (D/A; 10 Bit)
C 570 C (A/D; 8 Bit)
C 571 C (A/D; 10 Bit)
C 574 C (A/D; 12 Bit).

- **dynamische RAM-Schaltkreise**

Das Kombinat Carl Zeiss JENA stellte als Neuheit den 256 K x 1 Bit organisierten RAM in 1,5 µm-CMOS-Technologie **U 61256** (Bild 2) in den Geschwindigkeitsklassen 80 (z. B. U 61256 DC 08), 100, 120 und 150 ns sowie im Duroplast- und im Keramikgehäuse vor. Durch die Betriebsarten *statischer Seitenzugriff* und *schneller Page-Mode* ist ein hoher Datendurchsatz möglich. Das Kombinat Keramische Werke Hermsdorf bot einen hybriden 1-MBit-DRAM gleich in drei verschiedenen Versionen an. Der **4734** ist ein 128-KByte-Schaltkreis im 34poligen DIL-Gehäuse (Reihenabstand: 37,5 mm; Höhe: 4 mm) mit 16 Chips des NMOS-RAM U 2164 und einer Zugriffszeit von 260 ns. Die Schaltkreise **4735** und **4736** sind 256 K × 4 Bit und **1 M** × 1 Bit organisiert, befinden sich in einem 22poligen SIL-Gehäuse (Höhe: 12 mm; Breite: 5,5 mm) und vereinen 4 Chips des CMOS-RAM U 61256. Die Zugriffszeit beider Schaltkreise beträgt 155 ns.
Weitere Hybridspeicher sind der **8580** mit 64 KByte, 32poligem DIL-Hermetikgehäuse (Reihenabstand: 22,5 mm; Höhe: 8 mm), 8 Chips U 2164 und einer Zugriffszeit von 260 ns sowie der 4720X mit 64 K × 9 Bit, 32poligem SIL-Gehäuse (Höhe: 20 mm; Breite: 5,5 mm), 9 Chips U 2164 und einer Zugriffszeit von 260 ns (**47201**) bzw. 200 ns (**47202**).

- **statische RAM-Schaltkreise**

Neue statische CMOS-RAMs zeigte das Kombinat Carl Zeiss JENA. Der Schaltkreis **U 6264 DG** mit 8 KByte Speicherkapazität wurde mit Zugriffszeiten von 50 (U 6264 DG 05), 70 sowie 100 ns angeboten. Er ermöglicht Schlafbetrieb (power down) und ist pinkompatibel zum EPROM U 2764 C.
Mit einer Speicherkapazität von 2 KByte waren der **U 6516 DG 15** (maximale Zugriffszeit: 150 ns; maximale Schlafstromaufnahme: 100 µA), der **UL 6516 DG 15** (150 ns; 10 µA) sowie der **UL 6516 DG 25** (250 ns; 10 µA) vertreten. Sie besitzen einen Adreßlatch (sind deshalb nicht statisch betreibbar) und sind pinkompatibel zum EPROM U 2716 C.
Das Kombinat Keramische Werke Hermsdorf stellte den Hybridschaltkreis 8563X mit 4 KByte Speicherkapazität im 24poligen DIL-Hermetikgehäuse (Reihenabstand: 22,5 mm; Höhe: 8 mm) vor. Er vereint 8 Chips der Schaltkreisfamilie U 224. Der Schaltkreis **85631** beinhaltet die Chips US 224 X 20, hat eine Schlafstromaufnahme von 24 µA und eine Zugriffszeit von 210 ns. Die Schaltkreise **85632** und **85633** beinhalten Chips des UL 224 X 30 (240 µA; 310 ns) bzw. UL 224 X 35 (240 µA; 360 ns).

- **PROM-Schaltkreise**

Im Angebot des Kombinates Mikroelektronik befanden sich der 8-KByte-EPROM **U 2764 CC** im 28poligen Keramikgehäuse und seine PROM-Version **U 2664 DC** im Duroplastgehäuse (Bild 3). Diese NMOS-Schaltkreise können mit Zugriffszeiten von 250 (z. B. U 2764 CC 25), 350 und 450 ns geliefert werden.

Computertechnik

Die wachsende Bedeutung dieses Bereiches war auch auf der diesjährigen Leipziger Frühjahrsmesse deutlich zu erkennen: sei es als Mittel zur Rationalisierung von Leitung und Planung oder zur Lenkung der Produktion bis hin zur unmittelbaren Steuerung der Maschinen.
Um dem Leser den Überblick über die Vielfalt der angebotenen Computer etwas zu erleichtern, wollen wir im folgenden eine gewisse Klassifizierung vornehmen. Es muß jedoch darauf hingewiesen werden, daß es keine verbindliche Definition dafür gibt und sich die Grenzen – vor allem durch den technischen Fortschritt bedingt – ständig verschieben. Leistungen früherer Minirechner im Schrankformat werden heute oft von Auf- oder Untertisch-Arbeitsplatzcomputern erzielt; deren frühere Leistungen heute oft schon von Personalcomputern. Wenn man als einen Vergleichsparameter beispielsweise die Operationsgeschwindigkeit betrachtet, so erreichen die modernen PCs auf Basis des I 80386 heute mit bis zu vier MIPS (Millionen Instruktionen pro Sekunde) Geschwindigkeiten, die vor kurzem noch sogenannten Workstations, das heißt speziellen Ingenieurarbeitsplätzen vor allem für CAD-Aufgaben und meist mit Motorola-Prozessoren, vorbehalten waren. Diese wiederum sto-

ßen bereits in Bereiche von 9 und 10 MIPS vor. Dazu kommen die rasante Entwicklung der Speichertechnik, veränderte Architekturen und ein wachsendes Softwareangebot, die die Leistungen jeder Computerklasse ständig nach oben erweitern.

• Minicomputer

Schwerpunkt der Robotron-Exposition war zweifellos der 32-Bit-Rechner mit virtuellem Speicher (RVS) mit der Robotron-Bezeichnung Superminicomputer **K 1840** (Farbbild 1; alle Farbbilder siehe 2. Umschlagseite). Mit diesem Rechner werden die Leistungsgrenzen zu den kleinen ESER-Modellen erreicht bzw. hinsichtlich der Operationsgeschwindigkeit (1,1 Mio Op/s) sogar überschritten. Weitere Merkmale sind der Hauptspeicher mit 2 bis 16 MByte (auf der LFM mit 8 MByte ausgestattet), ein virtueller Adreßraum bis 4 GByte, bei Datenübertragungsraten von 2 MByte/s eine externe Speicherkapazität von max. 3 GByte und die Fähigkeit, mit dem K 1840 leistungsfähige Mehrnutzersysteme aufzubauen. Dazu dienen für echtzeitorientierte Aufgaben das Hauptbetriebssystem SVP 1800 und für interaktiven Teilnehmerbetrieb MUTOS 1800 (UNIX-kompatibel). Für den Aufbau lokaler und globaler Netze mit 8- und 16-Bit-PCs sind Netzroutinen vorhanden. Der K 1840 ermöglicht somit den Aufbau effektiver Komponenten für eine rechnerintegrierte Fertigung und für hohe Rationalisierungseffekte in den Bereichen CAD/CAM, CAQ (Qualitätssicherung) und CAP (Arbeitsplanung). In einer der nächsten MP-Ausgaben werden wir den K 1840 ausführlicher beschreiben.

Ebenfalls einen 32-Bit-Minirechner präsentierte die bulgarische Hanelsorganisation Isotimpex mit dem **ISOT 1055C** (Farbbild 2), der seit 1986 produziert wird. Er ist kompatibel zur VAX 11/730, hat einen Hauptspeicher von 5 MByte und einen virtuellen Adreßraum von 16 GByte. Das Plattenspeichersubsystem besteht aus dem Controller und zwei Plattenspeichern CM 5416 mit je 200 MByte. Als Betriebssystem wird MOS VP verwendet. In Leipzig war der ISOT 1055C zur Darstellung der Leistungsfähigkeit im CAD-Bereich mit der Grafikstation ISOT 1040C gekoppelt, deren Grafikprozessor ebenso wie der Prozessor des ISOT 1055C auf der Bit-Slice-Prozessor-Schaltkreisfamilie Am 2900 basiert. An die Grafikstation können zwei Arbeitsplätze mit einem Monitor hoher Auflösung (1024 × 1024 Pixel) und einer 60-Hz-Bildfrequenz (vertikal) angeschlossen werden.

Mit dem **ISOT 1054** zeigte Isotimpex einen weiteren Minirechner, basierend auf der 16-Bit-Prozessorfamilie M 16–3. Der Rechner ist vor allem für Echtzeitaufgaben vorgesehen, das entsprechende Betriebssystem ist das DOS RV–B Vers. 4 (RSX 11). Ein Programmpaket für Netzarbeit, SET 3, ermöglicht, im Rechnerverbund mit bis zu 16 Terminals zu arbeiten.

Die Kiewer Produktionsvereinigung Elektronmasch offerierte die Weiterentwicklung des 16-Bit-Minirechners CM 1800, den **CM 1810** (Farbbild 3). Gegenüber dem Vorgänger mit KR 5801-K80A-Schaltkreis besitzt der CM 1910 den Mikroprozessor K 1810 WM86 und erreicht damit eine Verarbeitungsgeschwindigkeit von 2 Millionen Op/s. Die Arbeitsspeicherkapazität beträgt 256 KByte, kann aber auf 1, 2, 3 oder 4 MByte ausgebaut werden. Der ROM läßt sich von 8 KByte auf 32 KByte erweitern. Als externe Speicher werden Floppy-Disks (CM 5640) mit 320 KByte, Zweiplatten-Wechselkassetten (CM 5408) mit 16 MByte und Festplatten (CM 5505) mit 20 MByte verwendet. Interessant ist beim CM 1810 die Variabilität hinsichtlich der äußeren Gestaltung. Die Anlage kann nämlich neben der im Bild gezeigten Schrankvariante auch als Auftisch- oder Untertischversion geliefert werden.

Erstmals in Leipzig gezeigt wurde vom tschechoslowakischen Außenhandelsunternehmen KOVO das 16-Bit-Mikrorechnersystem **M 16–22**, eine Version des Rechners CM 50/50 in Schrankausführung. Für viele Anwender wird es als Ablösung für die Anlagen CM 4/20 oder CM 52/11 von Interesse sein. Das Haupteinsatzgebiet wird in SKR-Netzen (unter Ethernet) und in der Prozeßsteuerung liegen. Der 0,5-MByte-Hauptspeicher ist auf 2 MByte erweitert; als Betriebssystem wird das DOS RV V3 verwendet, das Echtzeitbetrieb unterstützt.

Mit dem rumänischen Minirechner I 100 starteten die ICE-FELIX-Werke vor elf Jahren eine auch bei vielen DDR-Nutzern bekannte Entwicklung, die sich über den I 102 F fortsetzte und zur neuesten Version, der **I 106** (Farbbild 4) führte. Hauptmerkmal dieser auch in Bit-slice-Technik (von AMD) realisierten Anlage ist die Erweiterung des Hauptspeichers auf 4 MByte. Die bei den Vorgängern auf sieben Leiterkarten untergebrachte CPU beansprucht jetzt nur noch zwei Platinen.

Das Echtzeit-Multiprogramming-Betriebssystem MIC-PLUS besitzt volle Kompatibilität zum RSX–11 M PLUS V 2.1 und DECNET–11 PLUS V 1.1.

• Mikrocomputer

Dieser recht umfangreichen Klasse lassen sich sowohl transportable Computer zuordnen – zum Beispiel der EPSON PX 16, dessen interessantes Modulkonzept wir in unserem Heft 9 vorstellen werden, als auch die zahlreichen Büro- und Personalcomputer bis hin zum leistungsfähigen Ingenieurarbeitsplatz, beispielsweise der Workstation HP 9000.

Zunächst aber ein Wort zu den sogenannten Kompatiblen (siehe auch 3. und 4. Umschlagseite): Obwohl der von der Firma IBM 1981 herausgebrachte PC nicht der erste und auch nicht der leistungsstärkste seiner Klasse war, haben unter den Bedingungen des Marktes international zahlreiche Hersteller diesem Konzept mit eigenen – kompatiblen – Produkten Rechnung getragen, so daß schließlich aufgrund seiner Verbreitung der PC bzw. Kompatible und mit ihnen das Betriebssystem PC–DOS bzw. MS–DOS zum inoffiziellen Standard wurden. (Das führte sogar dazu, daß der eigentlich für eine Computerklasse geschaffene Begriff Personal Computer = Computer zur persönlichen Benutzung oft als Synonym für IBM-PC bzw. IBM-PC-kompatibel verwendet wird.) Dieser Trend erhielt noch Auftrieb durch die technische Weiterentwicklung des PC zum PC XT 1983 und zum PC AT im Jahre 1984 (AT = Advanced Technology), deutlich sichtbar auf den internationalen Messen.

Auf der LFM'88 hielt sich das Angebot von XT- und AT-Kompatiblen etwa die Waage. Die AT-Modelle bieten vor allem den Vorteil höherer Verarbeitungsgeschwindigkeit durch den Prozessor 80 286, der gegenüber dem 8088 nicht nur eine höhere Taktfrequenz besitzt, sondern bestimmte Aufgaben auch mit weniger Taktzyklen erledigt. Für viele CAD-Anwendungen oder Desktop Publishing haben ATs daher bessere Voraussetzungen. Allerdings lassen sich auch XTs durch Umrüsten mit 80 286-Prozessorkarten später prinzipiell auf AT-Niveau heben, mit einer 80 386-Prozessorkarte sogar auf 32-Bit-Niveau. Solche und „echte" 32-Bit-PCs wurden auf der LFM außer von der polnischen Firma ZIPO zwar von keinem Aussteller angeboten, waren aber bei mehreren im Lieferprogramm. (Über das Systemangebot von ZIPO und andere PC-Applikationen werden wir in MP 8/88 berichten).

International umstritten ist noch, ob bzw. wann sich das von IBM 1987 vorgestellte Personal System PS/2 zur Ablösung der PC-Modelle wiederum als Standard durchsetzen wird. (Über das PS/2 berichteten wir im Überblick bereits in MP 8/87.) In Leipzig waren die Modelle 30, 50 und 60 zu sehen. Die Firma Risto (bzw. Böwe Systemvertrieb GmbH) zeigte die Modelle 30 und 60 in Verbindung mit gelungenen Bildschirm-Arbeitsplätzen vor allem für CAD/CAM-Anwender. Das Modell 30 (Farbbild 5) mit 8086-Prozessor (8 MHz) ist als Einstiegsmodell in das PS/2 gedacht, auf dem unter PC DOS 3.3 die auf herkömmlichen PCs erstellte Anwendersoftware abgearbeitet werden kann. Hinderlich ist allerdings, daß alle PS/2-Modelle nur über 3 1/2-Zoll-Diskettenlaufwerke verfügen; 5 1/4-Zoll-Laufwerke müssen extern betrieben werden. Die Hauptspeicherkapazität beträgt 640 KByte bis 2,6 MByte, die Kapazität der Diskette 720 KByte, die der Festplatte 20 MByte.

Das Modell 50 (Farbbild 6), neben dem Modell 30 und Modell 60 von der österreichischen Vertriebsfirma IBM ROECE gezeigt, besitzt als leistungsstärkeres Gerät den 80 286-Prozessor mit 10 MHz und – im Gegensatz zum Modell 30 – bereits die neue Micro-Channel-Architektur, die einen bedeutend schnelleren Datentransfer ermöglichen soll. Weitere Daten: 1 bis 7 MByte Hauptspeicher, 3,5-Zoll-Floppylaufwerk mit 1,44 MByte, 20-MByte-Festplatte (80 ms Zugriffszeit). Als Standmodell ausgeführt ist das Modell 60 (Farbbild 7), das sich vor allem durch größere Speicherkapazität und mehr Erweiterungsmöglichkeiten auszeichnet. So läßt sich der Hauptspeicher bis zu 15 MByte aufrüsten, die Festplattenkapazität auf 70 MByte bei einer Zugriffszeit von 30 ms. Auch hier kommt der 80 286-Prozessor mit 10 MHz zum Einsatz. Wie das Modell 30, so können auch die Modelle 50 und 60 unter PC DOS 3.3 betrieben werden, letztere zusätzlich jedoch auch unter dem neuen Betriebssystem OS/2. Damit lassen sich nun auch die Multitasking-Eigenschaften des 80 286 in der geschützten Betriebsart (protected mode) nutzen. Das OS/2 ist Teil der IBM-System-Applikations-Architektur (SAA), einem Katalog von Regeln und Konventionen, deren Einhaltung es erlaubt, Anwendungsprogramme vom System /370 über die System /3X-Familie bis zu den PCs einsetzen zu können. Das leistungsstärkste Gerät des PS/2, das Modell 80 mit 32-Bit-Prozessor 80 386, war in Leipzig nicht ausgestellt.

Das sowjetische Außenhandelsunternehmen Elektronintrog zeigte mehrere Mikrocomputer, von denen vor allem der **MC 0104** in Tower-Ausführung bemerkenswert war (Farbbild 8). Er ist als Ingenieurarbeitsplatz vor allem für Echtzeitaufgaben und Multiuserbetrieb vorgesehen, wobei das Multifunktions-Betriebssystem MOS 32 M zum Einsatz kommt. Der 32-Bit-Mikroprozessor in VLSI-Technik KM 1807 WM2 erlaubt eine Rechengeschwindigkeit von 0,1 MIPS. Weitere Merkmale sind eine RAM-Größe von 1,5 MByte, ein physischer Adreßraum von 4 MByte, ein virtueller Adreßraum von 4 GByte. Als externe Speicher sollen 2 Diskettenlaufwerke mit je 800 KByte und eine Harddisk mit 20 bis 70 MByte eingesetzt werden können. Der Farbgrafikbildschirm Elektronika MS 7105 besitzt eine Auflösung von 512 × 560 Punkten.

Der ebenfalls gezeigte **MC 0507.04** (Farbbild 9) ist ein 16-Bit-PC mit etwa 0,1 MIPS, einem RAM von 1 MByte und insgesamt etwa 6 MByte Externspeicherkapazität. Als Zentralprozessor wird der KM 1801 WM3 verwendet, als Gleitkommaarithmetikprozessor der KM 1801 WM4.

Schließlich war im Angebot von Elektronintorg noch der Personalcomputer **Elektronika 85 (MC 0585)** zu sehen (Farbbild 10).

Er basiert auf 16-Bit-Mikroprozessoren der LSI-Schaltkreisserie 1811.

Der RAM beträgt 512 KByte, der ROM 16 KByte, der Adreßraum 4 MByte. Als Verarbeitungsgeschwindigkeit wurden 0,6 MIPS angegeben.

Der Elektronika 85 besitzt 2 Diskettenlaufwerke mit je 800 KByte und eine Harddisk mit 10 MByte. Der Farbmonitor hat eine Auflösung von 512 × 560 Punkten. Als Betriebssysteme sind PROS und FODOS/RAFOS vorhanden; es besteht Programmkompatibilität zu den Rechnern CM 4, CM 5, Elektronika 79 sowie EL-KA 60 und MC 1212. Über einen Adapter soll sich IBM-Kompatibilität erreichen lassen.

Hewlett Packard, seit vielen Jahren auf der Leipziger Messe vertreten, stellte diesmal das **Modell 310** der 9000er Serie aus (Farbbild 11). Die Basis dieser Workstation ist der Prozessor Motorola MC 68 010, der über eine interne 32-Bit-Architektur und externe 16- bzw. 24-Bit-Daten – bzw. Adreßbusse verfügt, mit 10 MHz getaktet ist und mit 0 Wait states arbeitet. Der vorhandene 1-MByte-RAM läßt sich auf 7,5 MByte aufrüsten; der virtuelle Speicherraum beträgt 16 MByte. Mit diesen Merkmalen können die Modelle der Serie 9000 vor allem für anspruchsvolle Ingenieurarbeiten verwendet werden. Dazu trägt eine leistungsfähige grafische Peripherie und die entsprechende Software bei. Das Standard-Betriebssystem ist das multi-tasking/multi-user-Betriebssystem HP–UX (eine Implementation des UNIX V.2); bei Einsatz eines Koprozessors ist jedoch auch die Arbeit unter DOS möglich.

Hans Weiß/Herbert Hemke











v. l. n. r.

EC 1841 *UdSSR*
XT-kompatibel · K-1810WM86-Prozessor (8086) · K-1810WM87-Koprozessor (8087) · 512 bis 640 KByte RAM · 720 MByte Floppy · 20 MByte Festplatte

Siemens PCD-2 *BRD*
AT-kompatibel · 80286-Prozessor mit 6/8 MHz · 512 KByte RAM · 720 KByte/1,2 MByte Floppy · 20/40 MByte Festplatte

ENSCH Entec 160 *Luxemburg*
XT-kompatibel · 8088-Prozessor mit 4,77/8 MHz · 640 KByte RAM · 360 KByte Floppy · 20 MByte Festplatte

ALR DART *Singapur*
AT-kompatibel · 80286-Prozessor mit 8/10 MHz · optional 80287-Koprozessor · 1 bis 2 MByte RAM · bis 1,2 MByte Floppy · 20 MByte Festplatte

EPSON PC AX 2 *Japan*
AT-kompatibel · 80286-Prozessor mit 8/10 MHz · 640 KByte bis 16 MByte RAM · 1,2 MByte Floppy · 20/40 MByte Festplatte

Commodore PC 1 *USA*
XT-kompatibel · 8088-Prozessor mit 4,77 MHz · 512 bis 640 KByte RAM · 360 KByte Floppy

PP 06 *ČSSR*
XT-kompatibel · 8088-Prozessor · 8087-Koprozessor · 256 bis 640 KByte RAM · 250 KByte Floppy · 20 MByte Festplatte

Iskra Partner AT *Jugoslawien*
AT-kompatibel · 80286-Prozessor mit 6/8 MHz · 1 MByte RAM · 1,2 MByte Floppy · 20/40 MByte Festplatte · optional 20 MByte Streamer

Elwro 801 AT *Polen*
AT-kompatibel · 80286-Prozessor mit 6/8 MHz · 512 bis 2 MByte RAM · 360 KByte bis 1,2 MByte Floppy · 20 MByte Festplatte

Atari PC 2 *USA*
XT-kompatibel · 8088/2-Prozessor mit 4,77/8 MHz · 512 bis 640 KByte RAM · 360 KByte Floppy · 30 MByte Festplatte

Mazovia *Polen*
XT-kompatibel · 8086-Prozessor · 8087-Koprozessor · 256 bis 640 KByte RAM · 360 KByte Floppy · 10/30 MByte Festplatte













ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 7
S. 193–224

VT 160 **Ungarn** 1
AT-kompatibel · 80286-Prozessor · 640 KByte bis 2 MByte RAM · 1,2 MByte Floppy · 20/40 MByte Festplatte

EC 1838 **Bulgarien** 2
AT-kompatibel · 80286-Prozessor mit 8/10 MHz · 80287-Koprozessor · 640 KByte bis 3 MByte · 1,2 MByte Floppy · 10/20 MByte Festplatte





Kompatible im Überblick

Wie alle großen internationalen Industriemessen, so ließ auch die Leipziger Frühjahrsmesse dieses Jahres einen der gegenwärtigen Trends im Computerbereich erkennen: Die Kompatiblen sind im Kommen. Nie zuvor waren in Leipzig so viele Aussteller mit Geräten dieser Computerklasse vertreten. Gemeint sind damit die Personalcomputer, die den PCs der Firma IBM mehr oder weniger ähneln. Die Kompatibilität – sprich Verträglichkeit – reicht dabei von MS-DOS-Kompatibilität, also der Möglichkeit, das De-Facto-Standard-Betriebssystem für Einplatzsysteme im 16-Bit-Bereich nutzen zu können, bis zur Steckkarten- also Hardware-Kompatibilität. Die angebotenen PCs verschiedener Hersteller haben demnach alle ihre Spezifika, auf die im Rahmen unseres Messeberichtes in diesem Heft kaum eingegangen werden kann. Die beiden letzten Umschlagseiten machen dennoch deutlich, wie sich das Konzept einer Computerklasse innerhalb weniger Jahre weltweit durchgesetzt hat.



POC 286 **Taiwan** 4
AT-kompatibel · 80286-Prozessor mit 16 MHz · 1,2 MByte Floppy · 40 MByte Festplatte

EC 1834 **DDR** 3
XT-kompatibel · K-1810WM86-Prozessor (8086) · 156 bis 640 KByte RAM · 320/720 KByte Floppy · 30/50 MByte Festplatte




ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK
MP
Heft 8 · 1988
Mikroprozessortechnik
VEB Verlag Technik Berlin
ISSN 0232-2892
UNIX
UNIX
UNIX

Bild 1 Struktur des UNIX-kompatiblen Betriebssystems Apollo DOMAIN/OS

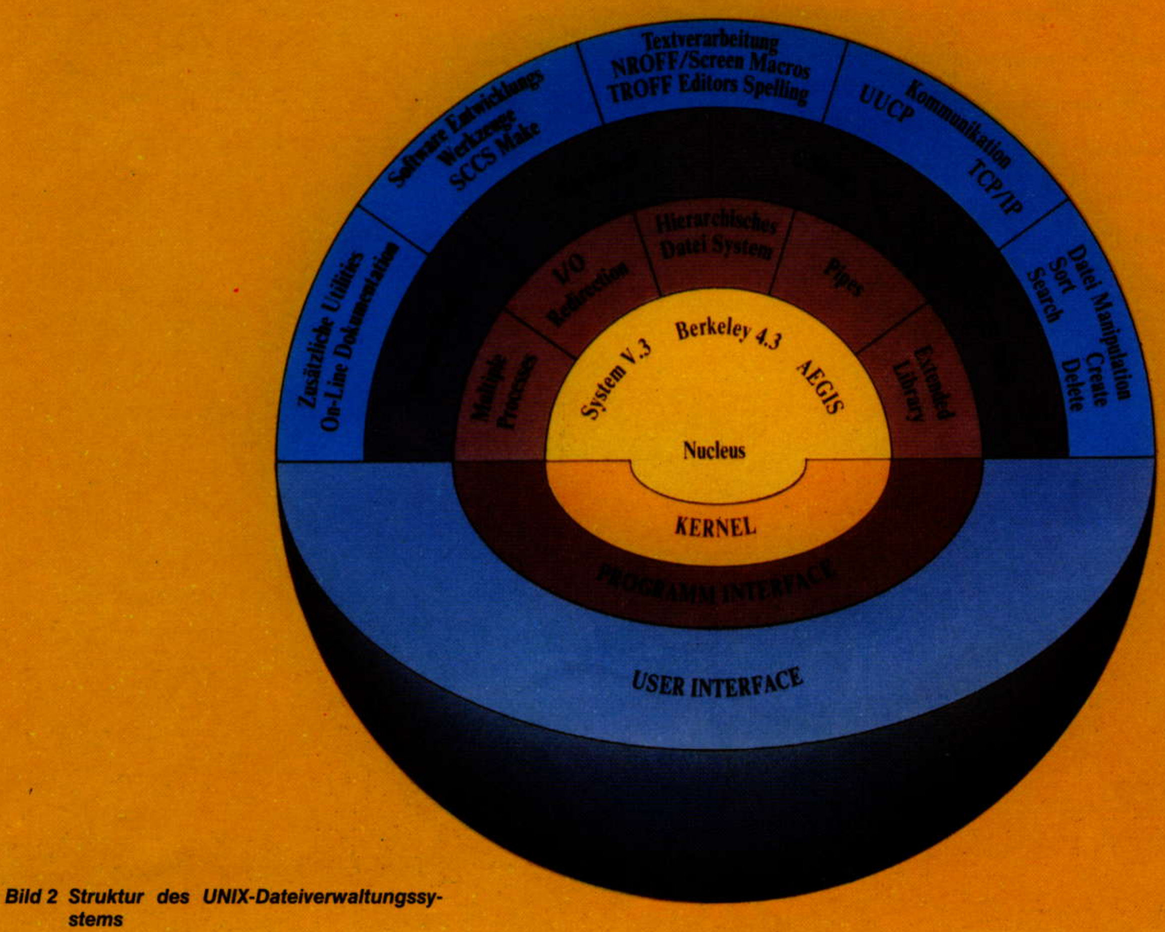
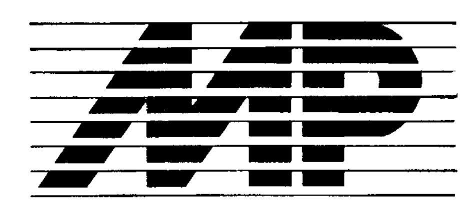


Bild 2 Struktur des UNIX-Dateiverwaltungssystems

- '/' root
  - unix ⋮
  - bin: adb, ar, as, bc, cat, cc, chmod, cmp, cp, date ⋮
  - etc: adduser, checklist, ddate, group, init, mount ⋮
  - dev: console, hd 0, hd 1, hd 2, hd 3, hd 4, mem, tty 1, tty 2 ⋮
  - lib: as2, c0, c1, c2, libc.a, libl. a, libm.a ⋮
  - last+found
  - usr
    - bin
    - man
      - man 0 ⋮
      - man 1: echo. 1, echo. 2, kill. 1, ld. 1 ⋮
      - man 2: dm, exp ⋮
      - man 3: get. s, put. s ⋮
    - include
  - tmp
  - anwender
    - peter
      - doc: konzept, doc.help ⋮
      - bin: help, guest, manager ⋮
    - ulli: programm 1, programm 2, hilfsdatei, spool, mkuser ⋮

Lesen Sie bitte dazu unseren Beitrag auf Seite 227.

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232-2892



Mikroprozessortechnik, Heft 8 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR-1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2870371); Herbert Hemke, Redakteur (Tel. 2870203); Sekretariat Tel. 2870381

Gestaltung Christina Bauer

Titel Tatjana Stephanowitz

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

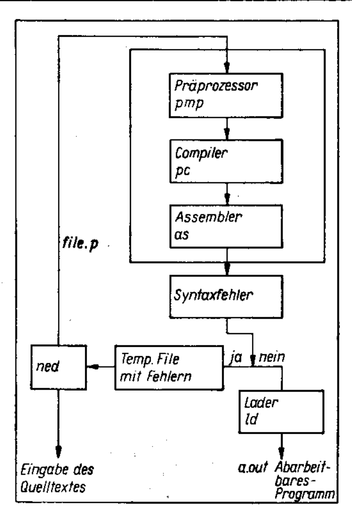
Redaktionsschluß: 20. Juni 1988

AN (EDV) 49837

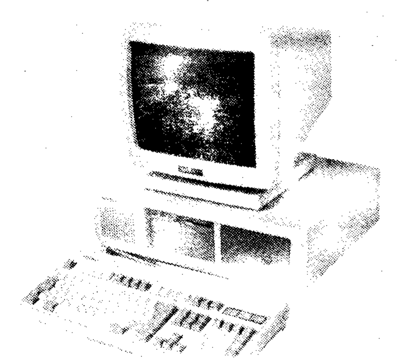
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

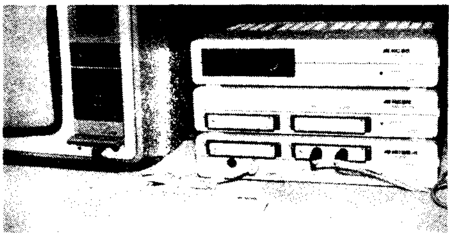
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tisku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Előfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Hà Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR-7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR-7010 Leipzig



Seite 229 (Bild 3)



Seite 234



Seite 253

Zum Titelbild

Da das Betriebssystem UNIX nicht an einen einzelnen Rechnertyp gebunden und leicht portierbar ist, wird von vielen Rechnerherstellern kein eigenes Betriebssystem entwickelt. UNIX stellt dabei das erste Betriebssystem dar, das alle Rechnerklassen (Mikrocomputer, Minicomputer und Großrechner) überstreicht. In unseren Beiträgen auf den Seiten 227 und 229 wird auf das Betriebssystemkonzept sowie auf Probleme der Softwareentwicklung mit UNIX eingegangen. Der Beitrag zum Betriebssystemkonzept beinhaltet u. a. eine Darstellung der Arbeiten zur Implementation UNIX-kompatibler Betriebssysteme auf DDR-Rechnern.

Inhalt

Leistungsanstieg in Sömmerda

Mit neuen Druckern und Personalcomputern erweitert das Büromaschinenwerk „Ernst Thälmann" Sömmerda in diesem Jahr sein Produktionsprogramm leistungsfähiger Ausrüstungen für die CAD/CAM-Technologie. Acht neue Erzeugnisse, darunter zwei elektronische Konsumgüter, wollen die Beschäftigten des größten Robotron-Betriebes in die Serienfertigung überleiten. Zu den Neuentwicklungen gehören der 16-Bit-Computer EC 1834 sowie der Laserdrucker EC 7230 (siehe unseren Messebericht). Für den neuen Drucker sind 15 Patente angemeldet. Das Betriebskollektiv sichert damit einen Neuheitsgrad seiner Produktion von mehr als 40 Prozent. Zahlreiche Initiativen der 13000 Werktätigen zielen auf eine weitere Erhöhung der Stückzahlen. So sollen 1988 erstmals mehr als 100000 Seriendrucker hergestellt werden. Das entspricht einem Zuwachs um mehr als 25 Prozent im Vergleich zu 1987. Voraussetzungen dafür werden gegenwärtig durch komplexe Rationalisierung der Druckermontage geschaffen.

ADN

EC 1057 für Csepel-Werke

Als erster Export-Auftrag ist im Juni in Budapest ein Robotron-Rechner der dritten ESER-Generation an das rechentechnische Unternehmen der Csepel-Werke übergeben worden. Gleichzeitig vereinbarten das DDR-Kombinat und der ungarische Betrieb eine langfristige Zusammenarbeit unter anderem auf dem Gebiet der Software-Entwicklung und Hardwareanpassung.
Robotron-Rechner haben sich in Ungarn unter anderem in den Videoton-Werken sowie an der Universität Debrecen bewährt.

ADN

Verhandlungen Japan–USA über Halbleiter-Exporte gescheitert

Zweitägige Verhandlungen Japan–USA über eine Neufestsetzung der Exportquoten für Halbleiter sind im Juni in Tokio gescheitert. Das wurde von Repräsentanten der Industrieverbände beider Länder auf getrennten Pressekonferenzen in der japanischen Hauptstadt mitgeteilt. Die USA-Elektronikindustrie fordert, ihren Marktanteil in Japan von gegenwärtig knapp zehn Prozent bis 1991 zu verdoppeln, da Japans Elektonikkonzerne in den USA denselben Marktanteil hätten.
Die japanischen Chiphersteller sehen dagegen in einer Exportgarantie für die USA-Halbleiterindustrie eine Einschränkung des freien Wettbewerbs. Sie schlagen stattdessen eine langfristige Kooperation von Chipproduzenten und Anwendern der USA, Japans und Westeuropas vor.

ADN

Was bedeutet COCOM?

Das sogenannte Koordinierungskomitee für Ost-West-Handelspolitik (COCOM) wurde 1949 gebildet. Ihm gehören gegenwärtig 16 kapitalistische Staaten, darunter alle NATO-Länder (außer Island) sowie Japan, an. Die Etablierung des Gremiums erfolgte unter Berufung auf westliche Sicherheitsinteressen mit dem Ziel, „strategisch bedeutsame Waren" und Technologien auflisten zu lassen, die von den COCOM-Mitgliedern nicht oder nur beschränkt in sozialistische Länder exportiert werden dürfen. Diese seit dem 1. Januar 1950 aufgestellten COCOM-Listen umfassen vor allem Geräte, Apparaturen und Ausrüstungen aus dem Bereich der Hochtechnologien – unter anderem Computer – jedoch auch, wie das 1978 gegen die UdSSR verfügte Röhren-Embargo bewies, Erzeugnisse der Schwerindustrie und des Maschinenbaus.
Der aus der Zeit des kalten Krieges stammende Export-Verbots-Index ist immer wieder ergänzt worden. Vor allem in den letzten Jahren haben die USA massivsten Druck gegen solche Staaten wie Australien und Neuseeland, neutrale europäische Länder wie Schweden, die Schweiz und Österreich sowie auch gerade auf Entwicklungsländer ausgeübt, das von COCOM verfügte Ausfuhrverbot zu respektieren.

aus Neues Deutschland vom 30./31. 1. 1988

In eigener Sache

Für eine weitere Redakteurstelle in unserer Zeitschrift suchen wir einen geeigneten Mitarbeiter bzw. eine Mitarbeiterin mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik.
Zu den Aufgaben gehören unter anderem das redaktionelle und fachliche Bearbeiten von Manuskripten, der Besuch und die Auswertung von Fachtagungen, -messen und -ausstellungen, die Zusammenarbeit mit Autoren und Gutachtern sowie ggf. das Testen und Beurteilen von Programmen, die der Redaktion zur Veröffentlichung eingereicht werden.
Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, rufen Sie uns unter Tel. 2870371 oder 2870203 an oder schreiben Sie an:
VEB Verlag Technik, Redaktion MP, Oranienburger Str. 13/14, Berlin, 1020

Hinweise für das Anfertigen von Manuskripten

1. Die mit Maschine einseitig beschriebenen Manuskripte – 2zeilig, 60 Anschläge/Zeile, 30 Zeilen/Seite – senden Sie bitte in zweifacher Ausfertigung ein.
Als Manuskript können auch Computerausdrucke mit einem sauberen und kontrastreichen Schriftbild dienen, sofern durchschnittlich 60 Anschläge je Zeile eingehalten werden (also wie bei Schreibmaschine **mit** Silbentrennung, kein Randausgleich!). Bei Verwendung von Matrixdruckern ist mindestens Near Letter Quality (NLQ) notwendig.
2. Der Umfang des Manuskriptes soll im allgemeinen 10 Seiten nicht überschreiten. Die maximale Länge des Textteiles von Angeboten und Suchmeldungen für die Rubrik „Börse" beträgt 30 Zeilen.
3. Das Manuskript beginnt mit einer aussagekräftigen, aber knappen Überschrift; es folgen der volle Vor- und Zuname, der akademische Grad und bei Beiträgen aus Ihrem Arbeitsbereich die Institution.
4. Versuchen Sie, dem Leser die Aufnahme der Informationen zu erleichtern; bei der Gestaltung des Textteiles sollten Sie also folgendes beachten:
– Beitrag übersichtlich gliedern!
(Zwischenüberschriften, Absätze; evtl. kurze Zusammenfassungen, Hervorhebung von Kernsätzen)
– Wo es der Aussage oder Anschaulichkeit dient, verwenden Sie bitte Bilder, Tafeln und Fotos (s. Pkt. 6).
5. Werden Abkürzungen verwendet, sind die im Duden aufgeführten zu nutzen; zusätzlich abzukürzende Begriffe sind beim ersten Auftreten auszuschreiben, und die Abkürzung ist in Klammern zu setzen.
6. Bilder, Tafeln und Fotos müssen als Anhang getrennt numeriert beigelegt werden. Die Einordnung ist im Text zu kennzeichnen. Für die erläuternden Unterschriften – sie sind bei allen Bildern, Tafeln und Fotos notwendig! – ist ein gesondertes Manuskriptblatt zu verwenden.
Bilder und Tafeln können als übersichtliche und eindeutig lesbare Bleistiftzeichnung bzw. maschinengeschrieben eingereicht werden. Um Satzfehler, insbesondere bei Listings zu vermeiden, sind diese als reprofähige Vorlage zu liefern (kontrastreicher Druck auf weißem Papier, auch saubere Schreibmaschinenschrift ist möglich).
Schwarzweißfotos sollten etwa das Format 13 cm × 18 cm haben (das Negativ wird nicht benötigt).
Farbfotos stimmen Sie bitte mit der Redaktion ab. Geben Sie bei honorarpflichtigen Fotos den Namen und die Adresse bzw. Konto-Nr. des Bildautors an!
7. Bedenken Sie, daß die Anzahl der Zeichen pro Zeile im Druck geringer ist als im Manuskript. Passen Sie also im Interesse einer korrekten Wiedergabe Gleichungen, Formeln, kurze Programmauszüge u. ä., die im laufenden Text erscheinen sollen, von vornherein an die **Spaltenbreite** (etwa 40 Anschläge) an.
Wo dieses nicht möglich ist, sind diese Zeilen als Bilder zu deklarieren.
8. Literaturverzeichnisse werden, in Schrägstriche eingeschlossen, fortlaufend numeriert, z. B. „wie in /1/ und /2/ eindeutig dargestellt wurde", oder „auch Meyer hat in /3/ darauf hingewiesen". Das Verzeichnis der verwendeten Literatur wird auf einem gesonderten Manuskriptblatt nach folgendem Schema angefertigt:

Literatur:
/1/ Menzer, R.; Richter, B.: Neue Technologien für Digitalisiergeräte. Feingerätetechnik, Berlin 34 (1985) 9, S. 386
/2/ Claßen, L.; Oefler, U.: Wissensspeicher Mikrorechnerprogrammierung. Berlin: VEB Verlag Technik 1986

9. Jedem Fachartikel ist gesondert ein etwa 10 Zeilen umfasssender **Referateteil** beizufügen, der das Wesentliche des Beitrages beinhaltet.
10. Bei Fachaufsätzen ist ein kurzes **Autorporträt** – mit Angaben über Alter, Ausbildung, beruflichen Werdegang, die jetzige Tätigkeit und Aufgabenschwerpunkte – erwünscht.
11. Unterstützen Sie bitte unsere Bemühungen, den Erfahrungsaustausch zu fördern, indem Sie für Interessenten eine Kontaktadresse und Telefonnummer angeben.
12. Denken Sie daran, bei Fachbeiträgen aus Ihrem Arbeitsbereich die Veröffentlichungsgenehmigung der Dienststelle beizulegen.
13. Nicht zu vergessen sind schließlich die private und dienstliche Anschrift, Ihre Konto-Nr. (bei mehreren Autoren Aufteilung des Honorars in Prozent angeben) sowie für evtl. Rückfragen eine Telefonnummer.
Bitte überprüfen Sie vor dem Absenden des Manuskriptes noch einmal die Einhaltung dieser Hinweise – Sie vermeiden damit unnötige Verzögerungen bei der Bearbeitung und Veröffentlichung Ihres Beitrages.

Das Betriebssystemkonzept UNIX

Dr. Ludwig Claßen
VEB Elektro-Apparate-Werke Berlin-Treptow „Friedrich Ebert", Zentrum für Forschung und Technologie

Die Entwicklung der Mikroprozessortechnik ist in den letzten Jahren durch eine enorme Leistungssteigerung auf dem Hardwaresektor gekennzeichnet, der für einen breiten Anwenderkreis, besonders bei den vielen dezentral eingesetzten Arbeitsplatzcomputersystemen, sichtbar wird.
Durch die Einführung moderner hochintegrierter Mikroprozessor- und Speicherchips (16-/32-Bit-MP; 256-KBit-/1-MBit-DRAM) sowie durch den Einsatz leistungsfähiger Massendatenspeicher (Floppy-Disk/Hard-Disk) mit großer Speicherkapazität ermöglichen Arbeitsplatzcomputer heute Problemlösungen, die früher nur Großrechnern vorbehalten waren: die Ausführung komplizierter wissenschaftlicher Berechnungen, die Lösung von Problemen, die bei der Leitung und Planung von ökonomischen Prozessen anfallen, die rechnergestützte Forschung, Entwicklung und Fertigung (CAD/CAM/CIM) und vieles andere mehr.

Einen entscheidenden Einfluß auf die effektive Nutzung der modernen Computersysteme haben das Betriebssystem und die zur Problemlösung eingesetzte Programmiersprache.
Auf diesem Gebiet hat in den letzten Jahren wohl kaum eine Entwicklung so eine Bedeutung erlangt, wie die außerordentlich breite Nutzung des Betriebssystemkonzeptes UNIX und der problemorientierten Sprache C /1/.
Die wichtigsten Faktoren, die international zu den mit zwei- und dreistelligen prozentualen Zuwachsraten pro Jahr steigenden Einsatzzahlen von UNIX-Computersystemen geführt haben, können in vier Punkten zusammengefaßt werden:

- UNIX wird nicht von einem einzelnen Hersteller beherrscht und ist nicht an einen einzelnen Rechnertyp gebunden. Diese Tatsache, zusammen mit der relativ einfachen Implementierung von UNIX, führte dazu, daß international viele Rechnerhersteller statt der sehr aufwendigen Entwicklung eines eigenen Betriebssystems für ihren neuen Mikrocomputer, zu dem leicht portierbaren UNIX griffen.
UNIX stellt dabei das erste Software-Betriebssystem in der Computergeschichte dar, das alle Rechnerklassen (Mikrocomputer; Minicomputer; Großrechner) überstreicht.
Dadurch entstand auf vielen unterschiedlichen Computersystemen eine nahezu einheitliche, herstellerunabhängige Benutzeroberfläche mit einer Fülle, auch wiederum relativ leicht portierbarer, Anwendungssoftware.
- UNIX bietet für die Softwareentwicklung eine hervorragend geeignete Arbeitsumgebung mit einer großen Anzahl von Entwicklungswerkzeugen (Tools), die durch ihren Aufbau und ihre Struktur sehr einfach und flexibel miteinander kombiniert werden können.
Die Grundkonzepte des Betriebssystems UNIX enthalten eine Reihe von genialen richtungweisenden Ideen.
UNIX ist deshalb ein sehr leistungsfähiges, umfangreich erprobtes Multi-User-/Time-Sharing-Betriebssystem, dessen Systemphilosophie modernsten Auffassungen und Anforderungen entspricht.
- Der Betriebssystemkern von UNIX besitzt einen relativ einfachen Aufbau, dessen, manchmal geäußerte, vermeintliche Primitivstruktur der Datei- und Prozeßverwaltungsmechanismen dem Anwender eine außerordentliche Flexibilität auf der Ebene der Systemaufrufe bietet.
- Zum Betriebssystem UNIX gehören als untrennbare Einheit die Programmiersprache C und der sehr komfortable Kommandointerpreter Shell. Beide besitzen eine sehr große Anpassungs- und Leistungsfähigkeit.
UNIX biete durch konsequent durchgängige Anwendung der maschinenunabhängigen Programmiersprache C und durch Vereinheitlichung der verbleibenden hardwarebezogenen Assemblerprogrammteile (Ein-/Ausgabe, Memory Management...) günstigste Voraussetzungen zur Implementation auf unterschiedlichsten Hardwarekonfigurationen.

Entwicklung des UNIX-Betriebssystemkonzeptes

Die Entwicklung des Betriebssystemkonzeptes UNIX nahm international im Jahre 1969/70 ihren Anfang. Ausgangspunkt war die Unzufriedenheit mit der vom Rechnerhersteller angebotenen Betriebssystemsoftware für einen Minicomputer. Zielstellung war die Schaffung eines universellen Betriebssystems für Kleinrechner mit standardisierter Benutzerschnittstelle. Außerdem sollte die Zugänglichkeit des Rechners für den Programmierer durch interaktiven Mehrbenutzerbetrieb erhöht und das koordinierte Zusammenwirken eines ganzen Teams von Softwarebearbeitern während einer Produktentwicklung unterstützt werden.
Die Arbeiten wurden zunächst vorwiegend auf der Basis von Minicomputern durchgeführt. Erreicht wurden die Zielstellungen durch die Einarbeitung einer ganzen Reihe von richtungweisenden neuen Basiskonzepten (hierarchische Dateiverwaltungsstruktur; Ein-/Ausgaberedirektion u. a. m.), durch eine hohe Modularisierung des gesamten Systems und durch die konsequente Anwendung der parallel entwickelten maschinenunabhängigen höheren Programmiersprache C. Etwa 95% der Systemsoftware von UNIX wurden in C geschrieben und nur 5% des auf einen konkreten Rechner Bezug nehmenden Teils von UNIX bedient sich der jeweiligen maschinenabhängigen Assemblersprache.
Das entstandene Betriebssystem, auf den Namen *UNIX* getauft, setzte sich Anfang der achtziger Jahre sehr schnell bei vielen Minicomputeranwendern durch, vor allem in Lehre und Forschung.
Das führte in der Folge zu einer Kettenreaktion, bezogen auf die Verbreitung dieses Softwaresystems.
Neue starke Impulse für die Applikation von UNIX und C gingen ab 1983 von der breiten Einführung sehr leistungsfähiger 16-Bit-Mikroprozessoren, hochintegrierter 64-KBit-Speicherchips und 5¹/₄-Zoll-Winchesterplattenlaufwerken aus.
Heute kann festgestellt werden, daß das Time-Sharing-Betriebssystem UNIX und seine maschinenunabhängige Implementierungsprogrammiersprache C zu einem Standard bei Klein- und Mikrorechnern geworden ist, der international von allen führenden Bauelemente- und Systemherstellern unterstützt wird.
Im Ergebnis dieser Entwicklung entstand und entsteht eine Fülle von in C für das Betriebssystem UNIX geschriebener Software, die für ganz unterschiedliche Aufgabenklassen Programme bereitstellt. Durch ihre Anpassung an eine einheitliche Systemschnittstelle (UNIX-Systemcalls) und durch ein einheitliches Sprachkonzept (C) ist diese Software auf ganz unterschiedlichen Rechnersystemen anwendbar.
Das Anwenderspektrum reicht heute von Prozeßsteuerungen durch UNIX-Mikrorechner über UNIX-gesteuerte Nachrichtennetze, bis zu CAD/CAM-Entwurfssystemen. Für Mikrorechnerentwicklungssysteme hat sich UNIX zum internationalen Standard schlechthin entwickelt.
Im Mittelpunkt der UNIX-Entwicklung stehen seit 1986/87 international insbesondere Vereinheitlichungs- und Standardisierungsbemühungen sowohl zum Systemkern als auch zu den Bibliotheksfunktionen und den Programmierwerkzeugen. Das ist besonders wichtig, nachdem einige Jahre verschiedene UNIX-Versionen auseinanderzulaufen schienen.
Das derzeit aktuelle Ergebnis dieser Standardisierung wird international unter dem Namen „UNIX System V Rel. 3" verbreitet.
An dieser Stelle noch einige Bemerkungen zu der etwas verwirrenden Namensgebung von UNIX-Betriebssystemprodukten. Es ist grundsätzlich zu beachten, daß der Name UNIX als Bezeichnung für ein Computerbetriebssystem nur für originale Produkte des die Urheberrechte besitzenden Rechtsträgers (Bell Systems Software /2/) zulässig ist. Wurde von diesem Rechtsträger eine UNIX-Weitergabelizenz an eine andere Firma oder Institution vergeben, durfte in der Vergangenheit der ursprüngliche Name nicht mehr benutzt werden. Das übernommene, notwendigenfalls etwas modifizierte und auf andere CPU-Systeme zugeschnittene UNIX muß einen neuen Namen bekommen (XENIX, OS/1, Onyx, MUNIX, SINIX ...; Bild 1, 2. US)
Eine andere Gruppe von UNIX-Anwendern arbeitet lizenzfrei mit eigenen Softwareprodukten, die mehr oder weniger 100%ig die originalen UNIX-Basiskonzepte in selbst geschriebenen Programmen übernehmen. Diese Betriebssysteme, meist als *UNIX-kompatibel* oder *UNIX like* bezeichnet, werden, bezogen auf das Vorbild, weder von Warenzeichen noch von Lizenzproblemen tangiert (COHERENT, IDIS, MUTOS, WEGA).

Standort von UNIX als Computerbetriebssystem

Die Eigenschaften eines Betriebssystems sind in ihren grundsätzlichen Funktionen für alle Rechnersysteme gleich. Sie unterscheiden sich ausschließlich in bestimmten Teilkomponenten, mit denen sie an spezielle Aufgabenkomplexe angepaßt werden.
Betriebssysteme lassen sich nach verschiedenen Gesichtspunkten systematisieren, sowohl in bezug auf die Anwendungsgebiete als auch in bezug auf ihre Arbeitsweise. Im Zusammenhang mit UNIX ist insbesondere

die Klasse der Multi-User-Betriebssysteme bedeutsam.
Multi-User-Betriebssysteme unterstützen parallel mehrere Anwender, die über unterschiedliche Bedienerterminals an ein zentrales Computersystem angeschlossen sind. Durch die Anwendung des Time-Sharing-Verfahrens, das jedem Anwender meist rein zyklisch, manchmal allerdings auch zyklisch prioritätsgesteuert, eine feste oder auch variable Zeitscheibe der CPU zuteilt, bekommt jeder Nutzer einen Teil der Rechenzeit des Systems zugewiesen. Auf Grund der hohen Rechengeschwindigkeit ist für den einzelnen Nutzer die Zeitteilung nicht oder kaum spürbar. Die übrigen Systemressourcen, wie Zugriff zum externen Massendatenspeicher, Druckerausgaben, Verfügbarkeit von bestimmten Ein-/Ausgabekanälen usw., werden prioritätsgerecht unter den Nutzern aufgeteilt.
Das Time-Sharing-Verfahren bildet die Grundlage für jedes Programm-Multitasking, bei dem mehrere Programme quasisimultan von einer CPU abgearbeitet werden. Es ist die Voraussetzung für ein Multi-User-Betriebssystem, wo mehrere Anwender quasisimultan, bei gemeinsamer Nutzung der Systemressourcen, an einer Rechnerzentraleinheit arbeiten können. Wichtig für die Anwendung von Multi-User-/Time-Sharing-Betriebssystemen sind hohe Rechenleistung der CPU (16-/32-Bit-MP), ein hinreichend großer Hauptspeicher (mindestens 512 KByte ... bis zu 4 MByte) sowie große und schnelle externe Direktzugriffsmassenspeicher (Hard-Disk), um die Programmabarbeitung und die Prozeßumschaltungen zeitlich zu optimieren.
UNIX wurde primär als Mult-User-/Time-Sharing-Betriebssystem konzipiert und entwikkelt.

Hauptmerkmale des Betriebssystems UNIX

Wird die Entwicklung auf dem Computersektor (Hardware und Software) einer Analyse unterzogen, so ist vor allem anderen die Tendenz zur Standardisierung kennzeichnend. Bei den Hardwaresystemen stehen dabei die Prozessortypen, die Bussysteme, parallele und serielle Interfaces, Leiterkartenformate usw. im Mittelpunkt.
Bei der Software ist die De-Facto-Standardisierung von Betriebssystemen, von Programmiersprachen, von Programminterfaces usw. unverkennbar.
Ursache dieser Entwicklung sind die Vorteile, die sich aus einer maximalen Austauschbarkeit und Koppelbarkeit von Hardware und Softwarekomponenten ergeben, sowohl innerhalb eines Rechners als auch zwischen Rechnern der gleichen oder einer anderen Klasse (Mikrorechner untereinander, aber auch Mikrorechner mit Minicomputern oder Großrechnern). Die bei der Software dadurch realisierbare Austauschbarkeit auf allen Ebenen,

- auf der **Programmebene** (Austausch von Anwenderprogrammen)
- auf der **Dateiebene** (Austausch oder gemeinsame Nutzung von Datenbeständen)
- auf der **Nachrichtenebene** (Austausch von Nachrichten /Messages/ zwischen Anwenderprogrammen in einem Rechner bzw. in Rechnernetzen)

spielt bei der überwiegenden Anzahl der Anwender heute eine außerordentlich wichtige Rolle.
Der Erfolg des Betriebssystems UNIX ist letztlich auch als ein Ergebnis dieser Entwicklung zu betrachten, stellt es doch ein umfangreiches Instrumentarium von universellen Unterstützungswerkzeugen für die Realisierung dieser Forderungen bereit.
Als allgemein hervorstechende Merkmale von UNIX können daneben insbesondere angesehen werden:

Hierarchische Dateiverwaltung

Das hierarchische, baumstrukturierte Dateiverwaltungssystem von UNIX (tree structured hierarchical file system) stellt das modernste existierende Dateiverwaltungskonzept dar. Es geht davon aus, daß jede Datei (file) Mitglied einer durch ein Dateiverzeichnis (directory) zusammengefaßten Gruppe von Dateien ist, die ihrerseits wieder selbst Teil einer übergeordneten Gruppe von Dateien sein kann. Die dadurch entstehende Baumstruktur von Dateien und Dateiverzeichnissen (beides wird vom UNIX-Dateiverwaltungssystem gleich behandelt) beginnt mit einem Wurzelverzeichnis (root directory) und ist auch oben und in der Breite nur durch die Hardwaresystemgrenzen beschränkt (Bild 2, 2. US).
Durch den vom Anwender in Ausdehnung und Struktur dynamisch definierbaren Dateibaum mit Directories und Subdirectories ist es möglich, beliebige Dateigruppen nach Zugriffsrechten, nach Projektzugehörigkeit, nach Anwendergruppen usw. aufzubauen und vom Betriebssystem zu verwalten.
Hinzu kommt, daß, neben der Gleichbehandlung von Dateien und Dateiverzeichnissen, eine Gleichbehandlung von Dateien und Ein-/Ausgabegeräten (bzw. von den Ein-/Ausgabegerätekanalprogrammen bzw. drivern) im UNIX-Dateiverwaltungssystem implementiert wurde.

Ein-/Ausgaberedirektion

Die Ein-/Ausgaberedirektion von Dateien, Geräten und Programmdaten basiert auf einheitlichen, untereinander kompatiblen Datenströmen. Sie ermöglicht durch einfachste Maßnahmen die Umleitung einer für ein bestimmtes Ein-/Ausgabegerät vorgesehenen Programmausgabe auf ein anderes Ausgabegerät oder auf eine neu anzulegende bzw. schon vorhandene Datei.

Pipes/Filter

Das Pipekonzept (pipe: Kanal) ermöglicht dem UNIX-Anwender, die Ausgabedaten eines Prozesses dynamisch als Eingabedaten für einen nächsten Prozeß zuzuweisen.
Programme, die, auf diesem Konzept aufbauend, einen vorgegebenen Eingabedatenstrom in einen neu erzeugten Ausgabedatenstrom umsetzen, werden unter UNIX als Filter bezeichnet.

Shell

Das Betriebssystem UNIX beinhaltet eine außerordentlich leistungsfähige Benutzerschnittstelle, den sog. Shell-Kommandointerpreter (shell: Schale). Dieses, wie eine Schale um den UNIX-Systemkern gelegte, Softwareinterface ermöglicht den Aufruf von Programmen – sowohl der Aufruf einzelner Programme als auch die parallele Bearbeitung mehrerer Programme und deren rekursiver Aufruf werden unterstützt. Es stellt dem UNIX-Nutzer einen Satz von Variablen und Kontrollstrukturen, ähnlich einer höheren Programmiersprache, zur Verfügung. Dadurch können zum Beispiel zu sogenannten Shellskripten zusammengestellte wiederkehrende Aufgaben automatisiert werden.
Das Shell-Benutzerinterface ist sowohl ein Kommandointerpreter als auch eine Programmiersprache in enger Verbindung zum Betriebssystem.
Vom Betriebssystem UNIX wird eine Shell-Standardvariante (Bourne-shell) bereitgestellt, die, wenn sinnvoll und notwendig, jederzeit vom Anwender durch eine eigene Benützerschnittstelle ersetzt werden kann.

C-Systemprogrammiersprache

Die Systemprogrammiersprache von UNIX ist die im Zusammenhang mit diesem Betriebssystem entstandene höhere Programmiersprache C. Sie kombiniert die Effizienz einer Assemblersprache mit den Steuerstrukturen moderner Hochsprachkonzepte. Die Programmerarbeitung unter UNIX ist allerdings wegen der Verfügbarkeit einer ganzen Reihe von Sprachcompilern nicht an die Sprache C gebunden. Trotzdem empfiehlt sich ihre Anwendung wegen des dann leichten Transportes der erarbeiteten Software von einem UNIX-System auf ein anderes (Softwareportabilität auf Quellprogrammebene). Für Systemprogramme sollte sie zwingend vorgeschrieben werden.
Neben diesen Hauptmerkmalen des Betriebssystems UNIX existierten eine Reihe von weiteren Besonderheiten, wie zum Beispiel die mögliche Aufteilung des Anwenderbereiches in leicht identifizierbare Gruppen oder die automatische Projektfortschreibung (SCCS: source code control system), die die besondere Unterstützung des Anwenders bei der Programmerarbeitung für ein großes Softwareprojekt absichern.
UNIX bietet außerdem eine ganze Kollektion modular aufgebauter, spezialisierter und effizienter Softwarewerkzeuge (tools) für die Programmentwicklung, die durch die Verfügbarkeit mehrerer Editoren, einer ganzen Reihe von Sprachcompilern (C, FORTRAN, PASCAL, COBOL, BASIC u. a.), von speziellen Programmtestunterstützungsmitteln (debuggern), Textverarbeitungssystemen und Datenbanksystemen u. a. m. ergänzt werden.
Es soll an dieser Stelle nicht verschwiegen werden, daß natürlich auch kritische Bemerkungen zum Betriebssystem UNIX existieren. Als Hauptpunkt ist dabei wohl die minimalistische Philosophie bei der Ein-/Ausgabekommunikation zu nennen. Mit dem Ziel, bei der Eingabe von Kommandos durch den Anwender mit möglichst wenigen Tastenanschlägen auszukommen und bei der Ausgabe von Meldungen durch das Betriebssystem auf einer Bildschirmseite ein Maximum an Informationen unterzubringen, wurden bei der Gestaltung des Ein-/Ausgabedialogs geradezu spartanische Befehls- und Kommandozeichenfolgen festgelegt. Das mag besonders den Anfänger etwas irritieren und akademisch erscheinen, ist aber für den fortgeschrittenen Nutzer bald zur Gewohnheit geworden. Voraussetzung für die effektive Arbeit mit dem Betriebssystem UNIX ist deshalb auch und besonders die exakte Beherrschung der Kommandosyntax und die sorgfältige Arbeit an dem Bedienterminal.

Vergleich von UNIX mit MS-DOS

Ein Vergleich unterschiedlicher Betriebssysteme wird immer stark vom Standort des Betrachters und vom Ziel des Vergleiches ab-

hängen. Es soll deshalb an dieser Stelle kein Vergleich zwischen UNIX und Betriebssystemen von Großrechnern oder Minicomputern folgen. Die hier seit längerer Zeit mit einer universellen Verbreitung vorhandenen Betriebssysteme werden durch UNIX keine existentielle Konkurrenz bekommen können, sondern nur an spezifischen Applikationsstellen eine Ergänzung erfahren.
Anders ist die Situation bei Mikrorechnern. Hier findet seit Jahren eine hauptsächlich von großen internationalen Rechnerherstellern geführte Auseinandersetzung um das dominierende Betriebssystem statt. Auf den in ihrer Leistungsfähigkeit durch die gegebenen Hardwaremöglichkeiten stark begrenzten 8-Bit-Mikrorechnern ist diese Auseinandersetzung klar zugunsten des Betriebssystems CP/M-80 ausgegangen.
Auf dem Gebiet der leistungsstarken 16-Bit-Mikrorechner existiert neben UNIX das Betriebssystem MS-DOS (oder PC-DOS) mit einem außerordentlich hohen Verbreitungsgrad.
Das Betriebssystem MS-DOS wurde für einen der ersten 16-Bit-Mikrorechner auf der Basis des Schaltkreises 8086 neu entwickelt. In vielem lehnte man sich bei der Konzeption von MS-DOS an das erfolgreiche CP/M an. Das Besondere an MS-DOS ist die Tatsache, daß ein großer internationaler Rechnerhersteller (IBM) dieses Betriebssystem für seinen in sehr großen Stückzahlen hergestellten 16-Bit-Personalcomputer als Softwarebasis ausgewählt hat. Die Unterstützung und Verbreitung, die MS-DOS von dieser Seite erfährt, ist deshalb außerordentlich groß. Die anfänglichen Probleme von MS-DOS wurden nach und nach beseitigt, und mit Beginn des Siegeszuges von UNIX wurden viele im vorhergehenden Abschnitt aufgeführten Hauptmerkmale von UNIX mit Ausnahme der Mehrplatzfähigkeit in fortgeschrittenen MS-DOS-Versionen übernommen. Das gilt für das hierarchische Dateiverwaltungssystem ebenso wie für die Ein-/Ausgaberedirektion, Pipestrukturen und das C-Sprachkonzept.
Bei dieser Entwicklung ist jedoch zu beachten, daß MS-DOS nach wie vor nicht UNIX-kompatibel ist. Für MS-DOS geschriebene Programme werden, auch wenn sie in der Programmiersprache C geschrieben wurden, nicht durch eine Neu-Compilation unter UNIX sofort lauffähig sein (z. B. unterschiedliche Systemaufrufe und verschiedene Ein-/Ausgabebibliotheken).
Die Qual der Wahl zwischen MS-DOS und UNIX bleibt bei dieser Entwicklung letztlich wohl keinem Anwender erspart. Als wesentliches Kriterium für den Einsatz von UNIX bleibt die Tatsache, daß MS-DOS auch in seiner allerneusten Weiterentwicklungsvariante (OS/2) immer ein Betriebssystem für ein Ein-Platz-Mikrorechnersystem bleibt und keine gleichberechtigten Multi-User-Fähigkeiten wie UNIX besitzt. Außerdem wird wohl keiner auf die Idee kommen, MS-DOS sowohl auf Mikrorechnern als auch auf Minicomputern und Großrechnern implementieren zu wollen.
Gerade durch die letztgenannte Eigenschaft von UNIX sichert aber dieses Betriebssystemkonzept dem Anwender auch langfristig seine umfangreichen Softwareinvestitionen.

UNIX in der DDR

Arbeiten zur Implementation UNIX-kompatibler Betriebssysteme auf in der DDR verfügbaren Großrechnern (ESER), Minicomputern (SKR) und Mikrorechnern (U 8000, K 1810 WM 86) wurden sehr frühzeitig etwa ab 1982 im

- Leitzentrum für Anwendungsforschung (LFA) Berlin
  PSU für ESER
- Technischen Universität Karl-Marx-Stadt Sektion Informatik
  VMX für ESER II
- VEB Kombinat ROBOTRON (RPD/RED) Dresden
  MUTOS 1600 für K 1600
  MUTOS 7100 für A 7100/A 7150,
  MUTOS 1834,
  MUTOS 1840
- Zentralinstitut für Kybernetik und Informationsprozesse (ZKI) Berlin
  MUTOS-U für BC 5120.16
- Zentrum für Forschung und Technologie des Kombinates EAW (ZFT-KEAW) Berlin
  WEGA für P 8000

durchgeführt.

Im Ergebnis dieser Arbeiten steht dem Anwender in der DDR heute eine abgestufte Palette von UNIX-kompatiblen Programmierumgebungen auf Rechnersystemen aller Leistungsklassen zur Verfügung.
Für alle Anwender ist dabei wichtig zu wissen, daß es langfristige Strategie im VEB Kombinat Robotron (dem größten Hersteller von Rechentechnik in der DDR) ist, für jedes produzierte Computersystem in Zukunft, zumindest als zweites lauffähiges Betriebssystem, auch eine UNIX-kompatible Benutzeroberfläche mit anzubieten.
Anders als beim VEB Kombinat Robotron, wo das UNIX-kompatible Betriebssystem meist als Ergänzung der Einsatzbreite zu einem doch dominierenden typenbezogenen Standardbetriebssystem anzusehen ist, hat man im KEAW Berlin den Weg eingeschlagen, das, inzwischen auch in großen Stückzahlen im Einsatz befindliche, Programmier- und Entwicklungssystem P 8000 ausschließlich mit dem voll UNIX-kompatiblen Betriebssystem WEGA auszurüsten.
Das System P 8000 stellt damit den einzigen in einer Großserie in der DDR hergestellten UNIX-kompatiblen Multi-User-Computer dar, der als Auftischgerät (maximal bis zu 8 Nutzer) einem breiten Anwenderkreis zur Verfügung steht /3/.
An dieser Orientierung bezüglich UNIX-kompatibler Betriebssysteme wird sicher sowohl im Kombinat Robotron (Zweitbetriebssystem) als auch im Kombinat EAW (Primärbetriebssystem) auch in Zukunft festgehalten werden, wenn man die z. Z. international ablaufende stürmische Entwicklung auf dem UNIX-Sektor berücksichtigt.

Literatur

/1/ Claßen, L.; Oefler, U.; UNIX und C – Ein Anwenderhandbuch. VEB Verlag Technik, Berlin 1987
/2/ UNIX Time-Sharing System. Bell System Technical Journal, July-August 1978, Vol. 57, No. 6, Part 2
/3/ Claßen, L.; P 8000 – ein universelles 16-Bit-Mikrorechnersystem. Mikroprozessortechnik 1 (1987) 3, S. 68
/4/ Tagungsbericht „3. Klausurberatung UNIX". Mikroprozessortechnik 2 (1988) 4, S. 128

✉ KONTAKT ☏

Zentrum für Forschung und Technologie des KEAW Berlin, Storkower Straße 101, Berlin, 1055; Tel. 4388730

Softwareentwicklung mit UNIX

Dr. Lothar Koch
Technische Hochschule Leipzig, Sektion Mathematik und Informatik

Bereits im Sommer 1985 wurde in der Konzeption zur Entwicklung der Informatik an der TH Leipzig eine durchgängige Nutzung des Betriebssystems UNIX (von 16 Bit Arbeitsplatztechnik bis ESER) festgelegt. Mit Beginn des Studienjahres 1987/88 konnten für alle Rechnerklassen (16-Bit-PC, CM 1420, EC 1056) UNIX-Systeme bereitgestellt werden. Auf der ESER-Anlage steht als verbindliches Ausbildungssystem das System VMX (virtual machine unix) zur Verfügung. Dieses System wird gemeinsam durch die TU Karl-Marx-Stadt, die TH Leipzig und die FSU Jena entwickelt und vertrieben. Für die Entwicklung und Erprobung einer Reihe von Softwarewerkzeugen wurden über mehrere Jahre Praktikanten der TU Dresden eingesetzt.

Entwicklungstrends zur Standardisierung von Betriebssystemen

Seit dem massenhaften Einsatz von Personalcomputern zu Beginn der 80er Jahre kann man zwei grundlegende Betriebsformen für die Nutzung von Computern unterscheiden:
- Einzelnutzer-Betrieb (single user)
- Mehrnutzer-Betrieb (multi user).

Für den professionellen Bereich (16-, 32-Bit-Technik) haben sich für diese beiden Betriebsformen die Betriebssysteme
- MS-DOS
- UNIX

durchgesetzt.
Bereits sehr zeitig gab es von seiten der Hersteller ernsthafte Bemühungen, die Vorteile beider Betriebsformen (gutes Antwortzeitverhalten bzw. zentrale Datenverwaltung und Organisation) zu vereinen. Besonders im kommerziellen Bereich, wo die PC-Technik am stärksten genutzt wird, kann auf eine zentrale Datenverwaltung nicht verzichtet werden. Deshalb orientieren die meisten Software- und Computerproduzenten für diesen Bereich immer deutlicher auf UNIX, wie eine Umfrage in /1/ bestätigt. Auch ausführliche Analysen der Entwicklung in den USA bestätigen diesen Trend. So findet man z. B. in /2/ folgende Aussagen:
MS/DOS... will become more and more like UNIX...
UNIX... will become more wide spread simply, because it is a product of very powerful corporation.
Dieser Entwicklung wird z. B. im System OS/2 und in der von Microsoft angekündigten MS/DOS Version 4.0 /3/ Rechnung getragen.
Viele Nutzer versuchen sich bereits UNIX-Möglichkeiten im MS-DOS zu erschließen. Davon zeugen z. B. allein 3 Artikel (Move-utility für MS-DOS (5/87), MS-DOS merkt sich 128 Befehle (4/87), MS-DOS mit Komfort (8/87)) in der Zeitschrift mc.
Um den Nutzern die Vorteile beider Betriebsformen zu bieten, bemühte sich eine Reihe von Firmen, entweder die ‚Mainframeumgebung' auf den PC zu bringen oder die Simula-

tion von z. B. MS-DOS unter UNIX/7/. Eine zukunftsorientierte Lösung ist die Bereitstellung von übergreifenden Gesamtsystemen, die natürlich eine Standardisierung voraussetzen. Mit der Lösung dieser Fragen begannen die führenden Universitäten der USA, unterstützt durch die marktbeherrschenden Firmen, bereits Anfang er 80er Jahre. 1982 wurde das System Athena auf der Basis des UNIX-Systems BSD 4.2 erfolgreich eingeführt /5/. Aufbauend auf diesen Forschungsergebnissen, wurde auf der National Computer Conference 1986 in Las Vegas von der Firma AT&T ein offenes System auf der Grundlage des UNIX-Systems V Release 3 vorgestellt /6/. Analog dem MIT-Projekt wurde 1984 für das Hochschulwesen der BRD mit dem Programm CIP (Computer Investitions Programm) begonnen, dessen Grundlage ebenfalls die Vernetzung von UNIX-Rechnern darstellt /3/.

Auch im Hochschulwesen der DDR stehen für alle Rechnerklassen ab 16 Bit Verarbeitungsbreite UNIX-Systeme zur Verfügung /7/. Beim Einsatz von UNIX-Systemen auf der verfügbaren Mikrorechnerbasis ist allerdings der hohe Speicherbedarf (operativ und extern) zu beachten. Hier seien zum Vergleich internationale Durchschnittswerte angegeben /8/.

	CP/M	MS-DOS	UNIX
Systemsoftware	100 K	250 K	8 M

Die zunehmende Bereitstellung von Festplatten ermöglicht den verstärkten Einsatz von UNIX. So wird z. B. an der TH Leipzig ab Herbstsemester 87 in einem 16-Bit-PC-Kabinett neben MS/DOS auch UNIX für die Ausbildung bereitgestellt.

Untersützt und abgestimmt werden die Entwicklungsarbeiten im MHF von der Arbeitsgruppe ‚Unifizierte Betriebssysteme'. Unter Beteiligung wichtiger Kombinate, der Akademie und des MHF konstituierte sich Ende 1986 ein UNIX-Arbeitskreis. Ziel dieses Arbeitskreises ist es, die UNIX-Entwicklung in der DDR in Richtung System V zu standardisieren (System V Interface Definition; x/open; posix) /8/. Für die verschiedenen Entwicklungslinien von UNIX sei hier auf den vorangehenden Beitrag und die inzwischen sehr umfangreiche UNIX-Literatur z. B. in /9/ verwiesen (Bild 1).

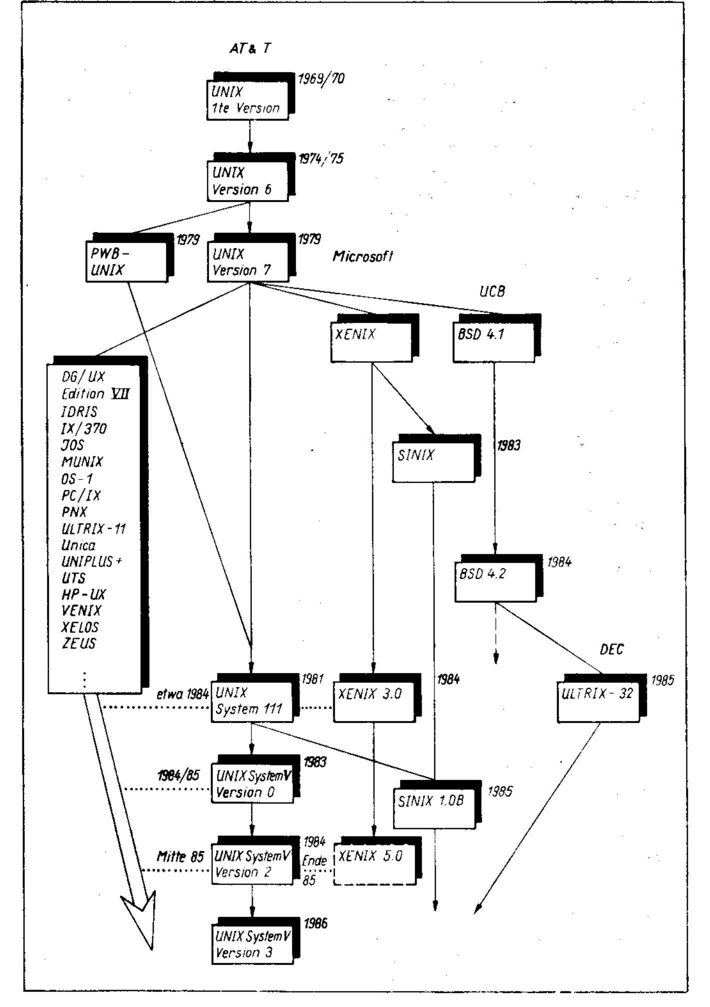
Aufbauend auf den Erfahrungen beim Einsatz der PSU in der Ausbildung, wurde im Rahmen der genannten Kooperationsgemeinschaft ein *echtes* UNIX-System für ESER-II-Anlagen – VMX – entwickelt. Dieses System wird ab Herbstsemester 87 als verbindliches Ausbildungssystem (Grundausbildung Informatik) auf einer EC 1 056 an der TH Leipzig genutzt. Im Unterschied etwa zur TU Karl-Marx-Stadt (Stand-Alone-Betrieb) erfolgt die Nutzung unter Steuerung von SVM (Bild 2). Neben der Kompatibilität zu UNIX besitzt VMX folgende wesentliche System-Merkmale:

- multiuser-timesharing-Betriebssystem,
- jeder Nutzer kann parallele Prozesse mit jeweils bis zu 16 MByte Adreßraum nutzen,
- Abbildung des virtuelen Speichers auf realen Speicher durch paging (eine Seite = 4KByte,)
- interne Verarbeitung im ASCII-Code,
- Filesystem mit Blöcken zu 4 KByte (d. h. auf 100 MByte Platten = 3 Blöcke/Spur).

Zur Einordnung von VMX in die UNIX-Familie sei hier auf /7/ verwiesen.

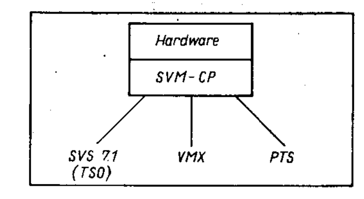
Überblick über Softwarewerkzeuge im VMX

Das Ziel der Entwicklung von UNIX in den Bell Laboratories war ein einfach handhabbares System für die Softwareentwicklung. Dem trägt ein umfangreicher Satz an Kommandos für die Programmentwicklung, insbesondere für die C-Umgebung, Rechnung. Ein weiterer wichtiger Gesichtspunkt bei Bell war, eine Entwicklungsumgebung für große Projekte zu schaffen, die einer unterschiedlichen Menge von Nutzern zur Verfügung steht. Ausdruck dafür war die Schaffung von PWB/UNIX (Programmers Work Bench) /10/. Ein Werkzeug aus dieser Umgebung ist z. B. das Quellcodeverwaltungssystem (SCCS). UNIX ist in immer stärkerem Maße die Basis allgemeiner Softwareentwicklungsumgebungen – sogenannter Systemproduktionsumgebungen /11/. Durch die Grundphilosophie von UNIX lassen sich darüber hinaus sehr effektiv die neuen Möglichkeiten der Hardwareentwicklung ausnutzen /12/ und offene Probleme – wie etwa in /13/ die fehlende Unterstützung von Windows und Bitmap-Grafik – schnell überwinden. In dem oben genannten Sinn bietet VMX noch keine ausgereifte und geschlossene Entwicklungsumgebung, schließt aber die PWB mit ein. Analoges kann zu den Hardwaremöglichkeiten gesagt werden. Das System nutzt alle Möglichkeiten der



◀ Bild 1 Entwicklungsrichtungen von UNIX nach /9/

Bild 2 Steuerung von VMX unter SVM ▶



Tafel 1 Einige Werkzeuge im VMX

Analyse	*lex*	*Erzeugung eines lexikalischen Analysators (scanner)*
	yacc	*Erzeugung eines syntaktischen Analysators (parser)*
Editoren	*ned*	*full-screen-Editor mit Sprachorientierung*
	ed	*Standardeditor*
	sed	*sequentieller Editor*
	hed	*hexadezimaler Editor*
Dateimanipulation (Beispiele)	*awk*	*Listenverarbeitung, Textmuster*
	grep, mgrep	*Mustersuche in Files*
	cmp	*Vergleichen von Files, Ausgabe der Unterschiede*
	sort	*Sortieren und Mischen*
Compiler	*cc*	*C-Compiler*
	f77	*Fortran-Compiler*
	pc	*Pascal-Compiler*
	qs	*Bildschirmsteuersprache auf Basis C (Quick-screen)*
	mcc, mf77, mpc, mqs	*Compiler in merge-Technik*
	as	*Assembler*
	ld	*Lader*
	sh	*shell*
Testwerkzeuge	*dcon*	*Dynamisches Testen (Prüfpunkte, Formatanzeige ...)*
	tcov	*Testen von Programmzweigen (Überdeckung)*
	mon	*Überwachung der Laufzeit*
	lint, mlint	*Überprüfung von C-Programm auf ‚saubere' Programmierung*
Programmwartung	*strip*	*Entfernen von Symboltabellen (nach Testung)*
	schart	*Aufbau von Strukturtabellen von C-Programm*
	cp	*Strukturierung von C-Programm*
	prettyp	*Formatieren von Pascalprogramm*
	pxref	*Pascal-Cross-Reference-Tabelle*
	make, mkmk	*Wartung zusammengehöriger Programme*
	sccs	*Quellcodeverwaltungssystem*
Dokumentation	*ms*	*Dokumentationsaufbereitung*
	nroff	*Textformatierung*

ESER-Anlage (EC 1056) einschließlich der Bildschirmtechnik (EC 7920), die allerdings modernen Anforderungen an Antwortzeitverhalten und Grafikfähigkeit nicht mehr gerecht werden.
Gemäß der Zielstellung von UNIX bietet VMX einen sehr umfangreichen Satz an Werkzeugen.
In Tafel 1 wird ein Einblick in die *wichtigsten* Werkzeuge gegeben.

Entwicklung von PASCAL-Programmen im VMX

Die am häufigsten genutzten Werkzeuge (ca. 80% der Sitzungszeiten) sind die Editoren. Leider wurde lange Zeit gerade diesen Werkzeugen zu wenig Aufmerksamkeit gewidmet. Mit **ned** steht im VMX ein bildschirmorientierter Editor zur Verfügung, der alle Möglichkeiten des Bildschirmsystems EC 7920 nutzt. Hier soll versucht werden, die Arbeitsweise kurz zu charakterisieren. ned baut bewußt auf dem Kommandosatz des Standardeditors **ed** auf. Der Aufruf erfolgt durch das Kommando

ned –optionen dateiname

Nach dem Aufruf wird der Bildschirm in drei Fenster eingeteilt. Die Grundform ist in Tafel 2 dargestellt (die Aufteilung kann durch Kommandos verändert werden).

Tafel 2 Aufteilung der Fenster für ned

<filename> *<messages>* *<altcount>*	*<no. lines>*
==⇒ PRIMAER KOMMANDOS	
TEXTBEREICH	ZEILEN NUMMERN *bzw.* ZEILENKOMMANDO BEREICH

Die Angaben in der Kopfzeile haben folgende Bedeutung:

<filename> – falls angegeben, Name der Datei
<massages> – Bereich für Fehlermeldungen
<alt.count> – Anzahl der Veränderungen nach letztem write
<no. lines> – Zeilenzahl

Der Editor verfügt über eine außerordentlich große Zahl von Kommandos. Für den Nutzer ist es wichtig, sich entsprechend seinen Anforderungen einen gut beherrschbaren Kommandosatz auszuwählen. Im Trivialfall läßt man sich den Text auf dem Bildschirm anzeigen, korrigiert den Text auf dem Schirm und schreibt den Schirminhalt wieder zurück. Die Primärkommandos müssen im festgelegten Bereich eingegeben werden (im Unterschied zu Vi). Trotz der Analogie zum ed muß die veränderte Arbeitsweise mit dem Bildschirm beachtet werden. Etwa der einzufügende Text wird zeilenweise (Zeilentrennung durch Wagenrücklauf <CR>) in den Textbereich eingegeben, und die letzte Zeile mit <ENTER> abgeschlossen. So entfällt z. B. die Eingabe eines zusätzlichen „." als *Endezeichen* für den Einfügemodus. Die beim ed angegebenen Kommandos behalten unter Beachtung dieses Gesichtspunktes ihre Bedeutung. Eine effektive Unterstützung bietet auch die Nutzung der Funktionstasten des Gerätes, denen ganze Kommandofolgen zugeordnet werden können. Hier eine Auswahl der Standardbelegung:

PF3 – Beenden der Editorarbeit mit vorheriger Rückspeicherung
PF7 – Textbereich nach oben
PF8 – Textbereich nach unten
PF10 – Textbereich nach links
PF11 – Textbereich nach rechts
PF12 – Kursor auf Primärkommandobereich

Die Zeilenkommandos dienen zur weiteren Erhöhung der Effektivität bei der Arbeit mit dem Bildschirm. Der Eingabebereich (bzw. Ausgabebereich für Zeilennummern) kann auch am linken Bildschirmrand festgelegt werden (Eingabe von Primärkommando D2). Im allgemeinen erstreckt sich die Wirkung eines Zeilenkommandos auf die Zeile, auf der das Kommando eingegeben wurde. Es gibt aber auch die Möglichkeit, die Wirkung auf eine ganze Zeilengruppe – einen Block – auszudehnen. Dazu muß das entsprechende Blockkommando meist am Anfang und am Ende des Blockes angegeben werden. Blockkommandos beziehen sich explizit auf die Zeilennummern (Bestimmung der Blocklänge). Alle auf einer Bildschirmseite eingegebenen Zeilenkommandos werden nach Eingabe von <ENTER> ausgeführt. Hier eine Auswahl:

[n] a Einfügen von n (n = 1 Standard) Leerzeilen nach der Zeile
[n] i Einfügen von n (n = 1) Leerzeilen vor der Zeile
[n] d Löschen dieser Zeilen, wobei dd als Blockkommando am
[d] d Anfang und Ende des Blockes stehen muß
s Teilt Zeile an Kursorposition (Newline)
[c] c Kopieren dieser Zeilen (des Blockes) auf die durch b (befor – bevor) oder t (target – danach) markierte Position (Zeile).

Neben der full-screen-Arbeit bietet ned auch die Möglichkeit der Sprachunterstützung. In sogenannten Sprachregelfiles können vorgefertigte Sprachkonstruktionen (Schlagworte) abgelegt werden. Im Fall von PASCAL stehen diese standardmäßig zur Verfügung. Erforderlich ist lediglich, für den Dateiindex p zu nutzen (bino.p). Gibt man z. B. beim Erfassen eines neuen PASCAL-Quelltextes im Textbereich

program <ENTER>

ein, erscheint auf dem Bildschirm als neues Bild eine vorgefertigte Programmstruktur

```
/*
:_
*/
program;
const
var
type
begin
end.
```

Dies erleichtert dem geübten ned-Nutzer eine effektive Eingabe von Programmtext, da für alle Anweisungen (etwa for, case, if) vorgefertigte Strukturen abrufbar sind.
Eine sehr effektive Arbeit bietet VMX auch für die Compilerläufe durch die sogenannte merge-Technik. Die merge-Compiler haben die Möglichkeit einer engen Zusammenarbeit mit dem Editor ned. Das System mischt die Fehlernachrichten des Compilers mit dem Quelltext in einem temporären File und übergibt dieses File dem automatisch aufgerufenen Editor. Nach erfolgter Korrektur werden die Fehlernachrichten im Quelltext wieder entfernt und eine erneute Übersetzung gestartet. Der Aufruf des merge-PASCAL-Compilers erfolgt durch das Kommando

mpc –optionen dateiname

Die Arbeitsweise des Compilers wird in Bild 3 (siehe Seite 225) deutlich.
Bei syntaktischen Fehlern wird das temporäre File als *ned-Bild* angezeigt. Die Fehlernachrichten bestehen aus einer Fehlernummer und einem Kommentar. Vor der Fehlernummer steht ein senkrechter Strich, der unter der Textposition steht, wo der Fehler bemerkt wurde (hat im allgemeinen nichts mit der realen Fehlerposition zu tun). Der Kommentar steht noch eine Zeile tiefer und ist durch einen „:" am Zeilenanfang gekennzeichnet. Der erste Fehler wird standardmäßig auf Zeile 4 mit einer stärkeren Helligkeit (intensiv) angezeigt. Weitere Fehlerzeilen erreicht man durch <PF5>. Nach Fehlerbeseitigung mit ned-Mitteln wird durch <PF3> wieder der Compiler aufgerufen. Es erscheint auf dem Bildschirm der Hinweis, daß der Compiler wieder arbeitet

:pc –c –E file.p

Wird kein Fehler mehr festgestellt, wird durch den Linker ein abarbeitbares File a.out erzeugt, welches durch

$ a.out

gestartet werden kann. Vor der weiteren Arbeit sollte man das File umbenennen, da beim Aufruf verschiedener Werkzeuge der Filename a.out genutzt wird. Eine Reihe von Werkzeugen dient der Unterstützung bei der Fehlersuche. Im VMX-PASCAL wird standardmäßig (option) ein Run time Error Reporting durchgeführt. Das heißt, daß bei Laufzeitfehlern oder bei Abbruch (<PA1>, <PA2>) des Programmes neben der Fehlermeldung ein traceback und ein symbolischer dump ausgegeben werden. Eine Verbindung zum Editor gibt es nicht. Im traceback kann man die Aufruffolge der Routinen bis zur Abbruchstelle, deren Adresse den Angaben in der Compilerliste – file.1 entspricht, verfolgen. Der symbolische Dump enthält die aktuelle Variablenliste. In Tafel 3 ist ein einfaches Beispiel dargestellt.

Tafel 3 Beispiel für Error Reporting

```
***ERROR*** floating divide exception (signal 8) received
Program terminated

at offset     282 in routine:     –pmain
==⇒ Routine – pmain
    Local variables
    bi = <uninitialited>        i = 0
    m = 4     n = 2
```


Literatur

/1/ UNIX/mail (1986) 2
/2/ Lewis: Operating System Standard of the 1985–95. IEEE Software (1985) 1
/3/ PC-Woche (1987) 28
/4/ Informationstechnik-it (1986) 4
/5/ Hausen, H. R.: Mikrocomputer in US-amerikanischer Hochschulausbildung. Angewandte Informatik (1984) 11, 12
/6/ Journal für Informationsverarbeitung – On-line (1986) 8
/7/ Apitz, N.: Koch, L.; Roth, M.: Entwicklung der programmtechnischen Basis an der TH Leipzig, abgeleitet aus dem internationalen Entwicklungsstand. Wissenschaftliche Berichte der THL (1987) 3
/8/ TOPIX (1987) 1–2
/9/ Gulbinus, J.: UNIX, Eine Einführung in Begriffe und Kommandos. Springer-Verlag, Berlin (West)-Heidelberg 1985
/10/ Mitze, R. W.: The UNIX System as a Software Engineering Environment. Software Engineering Environments, H. Hünke, North Holland 1980
/11/ Hruschka, P.: Integrierte Systemproduktionsumgebungen. Elektronische Rechenanlagen (1985) 2
/12/ Marty, R.: ET – An Editor Toolkit for Britmap-oriented Workstations, Tagungsberichte TH Karl-Marx-Stadt (1986) 10
/13/ Bach, F.: UNIX-Transparent. Springer-Verlag, Berlin (West)-Heidelberg 1985

✉ KONTAKT ☏

Technische Hochschule Leipzig, SMI, RZ, Karl-Liebknecht-Straße 132, Leipzig, 7030; Tel. 3 92 84 42

Datenbanken anlegen – aber wie?

Dr. Uwe Pape
Institut für Arzneimittelwesen der DDR

Ausgewählte Aspekte des Konzipierens, Aufbauens und Nutzens einer Datenbank

In der Formulierung einer Aufgabe ist immer ihre Lösung in bestimmtem Maße enthalten. Daher resultiert die intensivste Schubkraft für einen Fortschritt eigentlich nicht aus der Lösung, sondern aus der Formulierung *neuer* Aufgaben, weil es dabei gilt, *noch nicht Gedachtes zu denken.*

Bei der gegenwärtig und vielerorts fortschreitenden EDV-Einführung geht es im allgemeinen darum, solche neuen Aufgaben zu formulieren bzw. alte Verfahren zum Lösen von Aufgaben durch neue, bei denen die EDV eingesetzt wird, zu ersetzen. Vielen, die diese Vorhaben zu verantworten haben, sind sowohl das Leistungsvermögen des neuen Werkzeuges „EDV" als auch die zum Ausschöpfen dieses Vermögens erforderlichen organisatorischen Maßnahmen nicht genügend bekannt. Erschwerend kommt ferner hinzu, daß Hard- und Software nicht immer im erforderlichen Maße verfügbar sind.

Um die negativen Konsequenzen dieser Defizite möglichst gering zu halten, sind intensiv betriebene Weiterbildung und Zusammenarbeit der Informatiker mit den Fachleuten der jeweiligen Branche erforderlich. Es gelingt durch Anschauungsunterricht immer, vom Leistungsvermögen der Hard- und Software erste effektvolle Eindrücke zu vermitteln. Es ist aber eine ganz andere Sache, über diese Eindrücke hinausgehend den Willen zu wekken, sich des angebotenen Werkzeuges selbst zu bedienen. Ist das erste Eis beispielsweise mittels der allerorten so beliebten Textverarbeitungssysteme gebrochen, so mag das schrittweise Eindringen von Hard- und Software in wesentliche Dinge des Fachmannes noch gelingen.

Aber noch ist die allgemeine und wesentliche Beteiligung der EDV an den Problemlösungen in weiter Ferne, denn mit der Beteiligung der EDV an wesentlichen Dingen ist noch nicht ohne weiteres deren wesentliche Beteiligung an ihnen gegeben. Wenn beispielsweise Molekel dreidimensional und mehrfarbig an einem Bildschirm betrachtet und bewegt werden können, so ist hier die EDV an wesentlichen Dingen des Chemikers beteiligt. Deren wesentliche Beteiligung kommt aber erst zustande, wenn dem Chemiker darüber hinaus Vorschläge zur Lösung seiner analytischen oder synthetischen Aufgabe unterbreitet werden.

Allgemein betrachtet geht es hier um das Interface zwischen natürlicher und künstlicher Intelligenz. Dieser Bereich ist äußerst sensibel. Vor allem in den Gebieten mit nur wenig determinierten Vorgaben, wo also persönliche Erfahrungen, Fingerspitzengefühl, Meinungen und wohl auch Machtpositionen Einflüsse haben, dürfte es aus den verschiedensten Gründen (und das sind nicht nur die in der Sache selbst steckenden!) sehr schwer werden, diejenigen Probleme herauszufinden, die man zur Lösung durch die künstliche Intelligenz vorsehen kann. Da wird es auch wenig helfen, wenn man zu bedenken gibt, daß alle künstliche Intelligenz doch nur ein Abbild der natürlichen sein kann. Und schließlich geht es auch um neue Organisationsstrukturen mit tiefen Eingriffen in personelle und räumliche Beziehungen. Auf diesem Gebiet liegen wohl weltweit zu wenig Erfahrungen vor, so daß überall noch *Lehrgeld* zu zahlen sein wird.

Am leichtesten aber falsch ist es beispielsweise, *dieselben* Datensammlungen von nun an EDV-gestützt zu pflegen, die von alters her schon als Karteikarten vorliegen. Wenn Computer in vorhandene Arbeitsplätze eindringen, bilde man zu diesem Zweck die dort existierenden Karteikartensammlungen, ohne über die Abteilungsgrenzen hinauszusehen, „1:1" auf Disketten ab. So vorzugehen, ist bequem, erspart viele Streitgespräche und führt fatalerweise auch noch zu raschen Erfolgen. Aber auf diese Weise entstehen nur Kartenhäusern gleichende kurzlebige Erfolge! So kappt man von vornherein die Potenzen der EDV.

Im Zusammenhang mit der computergestützten Handhabung von Datensammlungen muß man zunächst die gewohnte individuelle oder auch gruppenorientierte Distanz zu ihnen vergrößern. Ebenso wie man nach Vergrößerung des Abstandes eine größere Fläche überschauen kann, ist es möglich, nach einer Vergrößerung des Abstandes zu Einzelfakten, eine größere Menge von ihnen zu überblicken. Mit den klassischen Hilfsmitteln und den menschlichen Sinnen geht von einem bestimmten Umfang einer Datenmenge an der Blick auf ihre Einzelheiten verloren. Aber auch hier besteht wieder eine Parallelität zu einer Abstandsvergrößerung im physischen Sinne. Dort verhilft ein Fernglas zur Wiedererkennbarkeit von Einzelheiten, hier dient ein Computer dem entsprechenden Zweck.

Ein Kriterium für den notwendigen Umfang einer Datenbank ist aus den logischen Zusammenhängen des gegebenen Datenmaterials abzuleiten. Dieses Kriterium ist mit der Entscheidung darüber gegeben, welche logischen Zusammenhänge in dem auf die Speichermedien abzubildenden Daten überhaupt enthalten sind. Die *überhaupt enthaltenen logischen Zusammenhänge* werden mit der Entscheidung darüber vorbestimmt, welche organisatorischen Struktureinheiten zugrunde gelegt werden sollen, ob also eine Arbeitsgruppe, eine Abteilung, ein Bereich, ein Direktorat, ein Betrieb, ein Kombinat oder gar ein Wirtschaftzweig sich am Aufbau und der Nutzung einer Datenbank beteiligen soll. Bezüglich des Konzipierens einer Datenbank kommt es immer *nur* darauf an, eine Datenbank als eine logische und nicht als eine physische Einheit anzusehen.

Oftmals ist es nicht möglich, die Informationen von den klassischen Datenträgern unmittelbar auf EDV-Speichermedien zu übertragen. In diesem Falle ist es üblich, Datenerfassungsblätter auszufüllen, d. h. in einer Vorstufe physische Datenerfassung mit klassischen Mitteln zu betreiben. Schon bei dieser Gelegenheit zeigt sich oft eine positive Nebenwirkung des Einführens der EDV. Es kommt zu einem *Gesundschrumpfen* oder zum Ausführen einer seit langem beabsichtigten Inventur.

Bei der physischen Datenerfassung muß vorausgesetzt werden, daß alle im gegebenen Datenmaterial vorkommenden Zeichen in den Zeichenvorräten des Computers und seiner peripheren Geräte enthalten sind. Leider bestehen noch immer Probleme bezüglich der Sonderzeichen in den verschiedenen lebenden Sprachen, die nicht wie die englische Sprache mit dem lateinischen Alphabet auskommen. Für die deutsche Schriftsprache geht es um die drei Umlaute in jeweils großer und kleiner Schreibweise und um das *ß*.

In der Schweiz schreibt man *ss* anstelle von *ß*. Wollte man diesen Schritt generell tun und bei der Gelegenheit die Umlaute gleich miterledigen, so waere doch zu fragen, ob es nicht graesslich fuer ein an die deutsche Schriftsprache gewoehntes Auge ist, Saetze lesen zu muessen wie beispielsweise:

„Haessliche aeussere Einfluesse verdriessen und stoeren haeuslichen Muessiggang!"

Die Absicht, auch in der deutschen Schriftsprache alle Wörter klein zu schreiben, besteht wohl inzwischen nicht mehr. Während man mit Hilfe dieser Eigenart den Intelligenzgrad der Software noch wird steigern können, fehlt ein gewichtiges Gegenargument, wenn es hieße, daß es ausschließlich eine Frage der Zeit sei, sich mit dem Tod der Umlaute und des *ß* abgefunden zu haben und daß es im übrigen ein Segen sei, endlich von der *Plage* befreit zu sein, mal *das* und mal *daß* schreiben zu müssen; es sei denn, man akzeptierte als wirksames Gegenargument, daß es schlicht zu viel verlangt ist, den Umgang mit Computern einerseits zu einem neuen Kulturgut werden zu lassen und andererseits an einem alten Kulturgut zu amputieren.

Es ist erforderlich, für eine Datenbank einen Verantwortlichen zu benennen, der als alleiniger Vertreter der Anwenderseite gegenüber der rechentechnischen Realisierung eines Datenbank-Projektes auftritt. Diese Forderung entspricht den Erkenntnissen über die notwendigen Voraussetzungen für einen aus der EDV-Anwendung maximal zu erzielenden Gewinn. Mit zunehmender Spannweite des logischen Rahmens einer Datenbank wird es aber immer schwieriger, die Einzelverantwortlichkeit zu garantieren, weil es wegen der Neuheit der Problematik in der gegenwärtigen Entwicklungsphase seitens der Anwender nur selten Mitarbeiter geben kann, die nicht nur über ein großes Breitenwissen in ihrer Disziplin verfügen, sondern auch noch hinreichende Informatik-Kenntnisse besitzen. Folglich wird es nötig sein, in einem Kollektiv (Datenbank-Administration DBA) Fachleute der jeweiligen Branche und Informatiker zu vereinen.

Um das Einhalten aller dort zu treffenden Verabredungen während der entscheidend wichtigen Phase des Aufbaues einer Datenbank, erstens überhaupt zu ermöglichen und zweitens kontrollieren zu können, muß man die DBA mit entsprechenden Kompetenzen ausstatten.

Bevor dieses Kollektiv zu seiner eigentlichen Arbeit kommt, bedarf es einer Etappe des Überwindens von Verständigungsschwierigkeiten durch gegenseitiges Eindringen in die jeweils andere Fachterminologie und Vorstellungswelt. Die verschiedenen Sichtweisen auf die gesamte Problematik müssen aufeinander abgestimmt werden. Sicher finden hier für Soziologen und Psychologen interessante Auseinandersetzungen statt. Konflikte treten auf, wenn es nicht gelingt,

sich im Wortsinne auseinander – also getrennt – zu setzen unter Wahrung des Blickes auf ein von allen gemeinsam zu lösendes Problem.
Zündstoff ergibt sich beispielsweise, wenn BASIC-Kenner als Software-Entwickler nach dem Motto „für jedes Problem ein neues Programm" verfahren und Ratschläge erteilen. Zündstoff ergibt sich ferner, wenn gelernte Software-Entwickler mit Neuerungen für die branchenspezifischen Fachleute glauben aufwarten zu müssen. Das sind Situationen, wie sie gewiß derzeit allerorten gegeben sind und die gegenwärtige Phase der technischen und gesellschaftlichen Entwicklung mitcharakterisieren.
Man kann nur hoffen, daß am Ende die der Informatik innewohnende integrierende und versachlichende Kraft zu einem alle Beteiligten befriedigenden Ergebnis führt, das dazu noch frei von schwächenden Einflüssen eines Kompromisses sein sollte.
Die EDV in einer dezentralen Organisationsstruktur zu nutzen, führt erst dann zum größten Erfolg, wenn gleichzeitig mit dem Übergang von der Zentralisierung zur Dezentralisierung der von der Dienstleistung zur Selbstbedienung stattfindet!
Problematisch ist es allerdings, wenn man die Forderung nach der Selbstbedienung nicht nur auf das Benutzen von Software sondern auch auf deren Produktion ausdehnt und zu diesem Zweck das postgraduale Erlernen einer Programmiersprache über Gebühr fördert oder fordert.
Auf Grund seiner leichten Erlernbarkeit kann z. B. BASIC zu einem Opiat werden und zu der Annahme verleiten, daß das Erlernen einer Programmiersprache identisch ist mit dem Erwerben von Kenntnissen zur Software-Entwicklung.
Kenntnisse dieser Art sind unabhängig von einer bestimmten Programmiersprache. Sie sind ein wesentlicher Inhalt der Fachdisziplin *Informatik*. Diese wissenschaftliche Disziplin ist noch sehr jung. Folglich sind Fachleute dieser Branche nur in geringer Anzahl verfügbar. Da aber in den nächsten Jahren der Bedarf an ihnen noch beträchtlich ansteigen wird, muß man erstens mit Rücksicht auf die nahe Zukunft die diesbezügliche graduierende Ausbildung fördern und zweitens in der Gegenwart nach einem optimalen Weg bei der Software-Produktion suchen.
Es ist sinnvoll, in jedem Falle vor dem Erzeugen neuer Programme zu prüfen, ob nicht mit den gegebenen Standard-Paketen Datenbankbetriebssystem, Textverarbeitungssystem und Kalkulationsprogramm ebenso zum Ziel zu kommen ist.
Im besonderen dürften in der gemeinsamen Nutzung dieser Standard-Software zur Lösung ein und desselben Problems oft nicht genutzte Potenzen verborgen sein. Um diese Potenzen möglichst restlos auszuschöpfen, sind spezielle Informatik-Kenntnisse erforderlich. Leider sind diese Kenntnisse nicht so leicht zu vermitteln wie ein paar BASIC-Kommandos. Diese Kenntnisse sind am besten aus der Anwendung zu gewinnen. Softwareanwendung setzt aber Kenntnisse voraus und zwar auch solche um deren Erwerb es geht. Der Anfänger kommt aus diesem Zirkelschluß nur heraus, wenn er EDV-Anwendung selbst betreibt, d. h. nach dem Motto *Lernen durch Handeln* verfährt. Dabei ist kollegiale Hilfe oder eine gute Dokumentation vonnöten. Wenn es dann doch unvermeidlich ist, neue Software zu produzieren, ist es nicht leicht, einen optimalen Weg dafür zu finden. Um die Fragen, wer produziert für welche Aufgaben womit Software, beantworten zu können, empfiehlt es sich, ein Kriterium aus der Anzahl der potentiellen Nutzer dieser Software und aus dem Umfang des mit ihr zu erschöpfenden Problemkreises, d. h. aus dem Grad ihrer Allgemeingültigkeit, abzuleiten. Ganz außer Frage steht es, daß Betriebssysteme, Assembler, Interpreter, Compiler oder Datenbankbetriebssysteme nicht von Programmier-Autodidakten zu entwickeln sind. Deren Betätigungsfeld ist die spezielle Anwender-Software.

Die EDV-Konzeption des Institutes für Arzneimittelwesen der DDR (IfAR)

In dieser Konzeption vom 26. 1. 1984 geht es im wesentlichen um eine dezentrale Hardware- und eine zentralisierte Software-Nutzung. Das endgültige Ziel besteht darin, ein Lokales Netz zu installieren.
Von Beginn an stand fest, keine EDV-Abteilung *klassischen Stils* einzurichten, d. h. keinen zentral zu nutzenden Computer aufzustellen, an den als periphere Geräte mehr oder weniger viele Bildschirm-Terminals angeschlossen sind. Inzwischen gibt es mehrere gute Gründe für diese Entscheidung. Zum damaligen Zeitpunkt stand die Tatsache im Vordergrund, daß die negativen Konsequenzen einer Havarie an Hard- oder Software bei Vorhandensein genau eines Computers sehr viel weitreichender sind, als wenn es in einem solchen Falle möglich ist, von einem Computer-Arbeitsplatz zum anderen zu wechseln.
Ähnlich bedeutsam für die Entscheidung zugunsten der Dezentalisierung war die Erfahrung, daß sich bei zentral installierten Anlagen und dem Aufbau von EDV-Abteilungen *neben* den Fachabteilungen nicht der an sich mögliche Gewinn aus der EDV-Anwendung erzielen läßt. An einem zentralen Computer entsteht notgedrungen ein mehr oder weniger großes selbständiges Kollektiv. Zwischen dessen Mitgliedern und denen in den Fachabteilungen kommt es meistens z. B. wegen organisatorischer Hürden nicht zu der erforderlichen Intensität der Kooperation.
Mit der Entwicklung der Bürocomputer und vor allem als dann ab 1986 der PC 1715 in großer Stückzahl produziert wurde, begann sich in der DDR das Prinzip der dezentralen Datenverarbeitung allgemein durchzusetzen.
Die IfAR-EDV-Konzeption ist von 1982 bis 1983 erarbeitet worden. Ab 1984 wurde damit begonnen, sie zu realisieren. Aus der Sicht der IfAR-spezifischen Aufgaben ging es in erster Linie darum, eine Datenbank aufzubauen. Als mit der physischen Datenerfassung begonnen werden konnte, stand im IfAR noch kein DBS zur Verfügung. Daher ist das DBS **Manus** entwickelt worden. Als konzeptionelles Schema wurde das CODDsche Relationenmodell gewählt.
Die Leistungsparameter von **Manus** waren zu einem bestimmten Teil durch die einer vorgegebenen Hard- und Software (spezieller BASIC-Interpreter) festgelegt. Es ist ferner vorausgesetzt worden, vor allem zum Aufbau einer Datenbank in der Lage zu sein sowie die von EDV-Erstanwendern zu überwindende Einstiegshürde möglichst niedrig zu halten. Die wesentlichen Merkmale von **Manus** sind:
● Es existiert im wesentlichen nur der Datentyp **Zeichenkette**. Eine Zeichenkette kann maximal 32 KByte lang sein. (Alle folgenden quantitativen Aussagen gelten unter Beachten dieses Parameters.)
● Beim Erzeugen einer Datei wird vom Nutzer nicht verlangt, einen Datentyp festzulegen, dem die Elemente eines Feldes zu entsprechen haben.
● Es ist nicht nötig, für die Längen von Zeichenketten als Elemente eines Feldes einen Höchstwert festzulegen.
● Die Anzahl der Feldnamen je Record ist unbegrenzt.
● Die Anzahl der Records je Datei ist unbegrenzt.
● Ein Feldelement kann im besonderen eine *implizite Tabelle* sein. Implizite Tabellen sind Datenstrukturen, die bekanntlich auch als *genestete* Tabellen oder als *Nest-Relationen* bezeichnet werden.
● Der Dialog ist in jeder der Komponenten von **Manus** durch eine feste Menü-Steuerung programminitiiert.
● Mit den Dateien sind neben den trivialen Handhabungen des Erzeugens, Korrigierens sowie des Transfers alle Manipulationen möglich, die sich auf die von E.F.CODD so genannten Operationen **Projektion, Join** und **Restriktion** zurückführen lassen.
● Arithmetische Operationen sind in **Manus** nicht möglich.

✉ KONTAKT ☎

Institut für Arzneimittelwesen, Große Seestraße 4, Berlin, 1120; Tel. 3651308

TERMINE

Internationales Messesymposium „Messen und Prüfen in der chemischen Industrie"

WER? Wissenschaftlich-Technische Gesellschaft für Meß- und Automatisierungstechnik der KDT
WANN? 8./9. September 1988
WO? Leipzig
WAS?
● Prozeßautomatisierung
● Sensoren und Sensorsysteme
● Chromatografie sowie pH-Wert-Messung
● Flüssigkeits- und Gasanalyse
● Spektralanalyse
WIE? Teilnahmewünsche bitte an Kammer der Technik, Präsidium, WGMA, Clara-Zetkin-Straße 115–117, Berlin, 1086

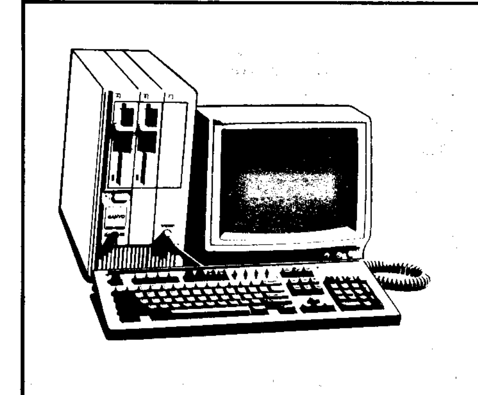
Müller

1. DDR-offene MIDI-Informationsbörse

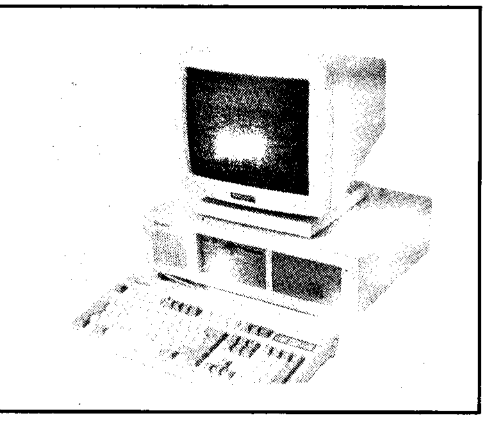
WER? Kulturhaus im Ernst-Thälmann-Park
WANN? 30. Oktober 1988, 10–18 Uhr
WO? Kulturhaus im Ernst-Thälmann-Park Berlin
WAS? Rund um den MIDI-Standard (Computerschnittstelle für Musikinstrumente) wird informiert, vorgetragen und ausgetauscht.
Aus dem Angebot:
– MIDI-Grundlagen
– MIDI und Licht
– MIDI mit DDR-Rechnern
– MIDI im Einsatz u. v. a. m.
WIE? Schriftliche Anmeldung unter dem Kennwort MIDI an: Kulturhaus im Ernst-Thälmann-Park, Computerclub, Dimitroffstraße 101, Berlin, 1055

Gätcke

Der IBM PC und seine Kompatiblen



Ob XT- oder AT-kompatibel, fast alle Hersteller übernehmen vom IBM-Vorbild auch den grundsätzlichen Aufbau ihrer PCs – kompaktes Rechnergehäuse, das sich neuerdings oft als Tower senkrecht stellen läßt, abgesetzte Tastatur und separater Monitor. Die Klasse der PCs ist damit meist schon äußerlich erkennbar – wie es drinnen aussieht, schildert dieser Beitrag zum IBM PC und seinen Kompatiblen.



Joachim Geiler, Matthias Wermann
Ingenieurhochschule Mittweida, Sektion Informationselektronik

1981 erschien mit dem IBM PC ein Personalcomputer auf dem Markt, der das Grundmodell für eine Rechnerfamilie werden sollte, die sich weltweit als Industriestandard durchgesetzt hat. Die wesentlichsten Vertreter dieser Familie sind der IBM PC/XT als kleineres Modell und der IBM PC/AT. Neben den Originalmodellen der Firma IBM erschien (und erscheint noch) auf dem Weltmarkt eine nahezu unübersehbare Vielzahl von Kompatiblen (siehe MP 7/88: 3. und 4. US), die in ihrer Ausstattung und Leistungsfähigkeit ihre „Vorbilder" teilweise übertreffen. Obwohl IBM Anfang 1987 mit dem Personal System/2 eine neue Rechnerfamilie vorgestellt hat /1/, dürfte das PC-Konzept noch eine Weile Bestand haben. Dafür spricht auch, daß alle Modelle des PS/2 als softwarekompatibel zum alten PC betrachtet werden können /2/ (siehe MP 7/88: Bilder 5, 6 und 7; 2. Umschlagseite).

Grundsätzlicher Aufbau

● IBM PC/XT und Kompatible

Hardwaremäßig besteht der PC aus einer Systemplatine (Motherboard), auf der sich neben CPU und Hauptspeicher eine Reihe von Steckplätzen für Ergänzungsbaugruppen befinden. Der Original-XT ist mit dem INTEL-prozessor 8088 mit 8-Bit-Datenbus ausgerüstet, der mit 4,77 MHz getaktet wird. Kompatible setzen häufig andere Prozessoren, wie den 8086 mit 16-Bit-Datenbus (z. B. im Schneider-PC), den V20 der japanischen Firma NEC (z. B. im EPSON PC+) oder den 80186, ein /2/. In Verbindung mit einer höheren Taktfrequenz (8 MHz, bei einigen Rechneren durch Knopfdruck auch während des Betriebes umschaltbar) wird dadurch eine gegenüber dem XT höhere Verarbeitungsleistung erreicht. Auf der Systemplatine befindet sich in den meisten Fällen auch eine Steckfassung für die Nachrüstung eines Arithmetikprozessors 8087. Beim XT befinden sich auf der Systemplatine bis zu 256 KByte RAM (64 KBit-Chips). Während hier Speichererweiterungen eine Erweiterungskarte nötig machen, sind bei neueren Kompatiblen häufig 512 KByte bis zu den maximal möglichen 640 KByte auf der Grundplatine vorhanden, wobei eine Fehlererkennung durch Paritätsbit üblich ist. Auf der Systemplatine befindet sich weiterhin ROM-Speicher im Umfang von etwa 32 KByte. Dieser ROM enthält den Anfangslader, Diagnoseroutinen, das sogenannte ROM-BIOS und (nur beim Original-XT) ein ROM-BASIC. Der Prozessor 8088/8086 besitzt einen 20-Bit-Adreßbus, kann also maximal 1 MByte Speicher adressieren.

In den Erweiterungssteckplätzen des Rechners sind neben der Speichererweiterung die Controller für Disketten und Festplatte zu finden. Als Diskettenlaufwerke werden bei den XT-Rechnern fast ausschließlich doppelseitige 5¼-Zoll-Laufwerke mit 40 Spuren pro Seite verwendet. Mit dem für MS-DOS üblichen Diskettenformat (9 Sektoren zu 512 Byte) ergibt das eine Diskettenkapazität von 360 KByte. Bei einigen Kompatiblen findet man auch Laufwerke mit 80 Spuren und einer Kapazität von 720 KByte. Da ein Großteil der heute zur Verfügung stehenden Programmpakete einen erheblichen Speicherplatzbedarf hat, ist der Einsatz einer Festplatte mit einer Kapazität von 10 MByte und mehr sinnvoll. Beim XT werden heute vorzugsweise Festplatten mit 20 ... 30 MByte und einer Zugriffszeit von 60 ... 80 ms verwendet.

Weitere Erweiterungssteckplätze werden für den Druckeranschluß (i. allg. Parallelschnittstelle nach dem Centronics-Standard) und eine Platte für die serielle Kommunikation (enthält meist zwei V.24-Schnittstellen) benötigt. Die außerdem vorhande Bildschirmkarte wird weiter unten beschrieben.

● IBM PC/AT und Kompatible

In seinem grundsätzlichem Aufbau (Systemplatine mit Erweiterungssteckplätzen) ähnelt der IBM PC/AT dem XT. Im AT wird im Gegensatz zum XT der Mikroprozessor 80286 eingesetzt. Dieser Prozessor kann in zwei Modi betrieben werden /3/. Im sogenannten *Real-Address-Modus* ist er zum 8086 kompatibel und kann 8086-Programme unverändert abarbeiten. Es steht wie beim 8086 ein Adreßraum von 1 MByte zur Verfügung. Im *Protected-Modus* können 16 MByte physischer Adreßraum verwendet werden (24 Adreßleitungen), und es steht eine Reihe von Hardwareunterstützungen für Multitasking und virtuelle Speicherung (1 GByte virtueller Adreßraum pro Task!) zur Verfügung. Unter dem (auch beim AT meistverwendeten) Betriebssystem PC-DOS bzw. MS-DOS wird der AT jedoch ausschließlich im Real-Address-Modus betrieben. Ein Teil der Fähigkeiten des 80286 bleibt damit ungenutzt. Auch im AT kann ein Arithmetikprozessor eingesetzt werden (80287). Der AT wird mit 6 bis 10 MHz Takt betrieben. Damit und mit der grundätzlich höheren Leistung des 80286 (z. B. Register-Register-Division in 22 Takten /3/ gegenüber 144 beim 8086 /4/) erreicht der AT besonders bei numerischen Anwendungen eine gegenüber dem XT um den Faktor 3 ... 5 höhere Leistungsfähigkeit. Nach /5/ bringt der Einsatz eines Arithmetikprozessors im AT eine prozentual geringere Verbesserung als in einem 8088-System.

Der AT verwendet im Gegensatz zum XT Diskettenlaufwerke höherer Kapazität. Zum Einsatz kommen sogenannte High-Density-Laufwerke, die bei doppelseitiger Nutzung und 80 Spuren pro Seite eine Nettokapazität von 1,2 MByte aufweisen (bei 15 Sektoren pro Spur). Für diese Laufwerke werden spezielle Disketten benötigt. Normale 360-KByte-Disketten können jedoch problemlos gelesen und auch beschrieben werden. Das Lesen von auf dem AT beschriebenen 360-KByte-Normaldisketten auf einem XT kann jedoch zu den auch vom Datenaustausch zwischen PC 1715 und A 5120 bekannten Problemen führen.

Beim AT und Kompatiblen kann eine Festplatte als Standard angesehen werden, nach /6/ kommen Platten mit Kapazitäten von 20 ... 80 MByte zum Einsatz, die sich gegenüber den beim XT eingesetzten auch durch eine geringere Zugriffszeit auszeichnen. Zur Drucker- und seriellen Karte gilt das zum XT Gesagte. Neben den XT-kompatiblen Erweiterungssteckplätzen sind auch spezielle AT-Erweiterungsplätze (16-Bit-Datenbus, 24-Bit-Adreßbus) vorhanden. Diese werden z. B. für Speichererweiterungen über die 640 KByte hinaus benutzt (sog. Extended Memory, wird unter MS-DOS jedoch nur als RAM-Floppy verwendet).

Bildschirmansteuerungen

Im IBM PC und seinen Kompatiblen werden unterschiedliche Bildschirmansteuerkarten eingesetzt, deren wichtigste im folgenden beschrieben werden /7/. Für die Darstellung von alphanumerischen Zeichen wird ein erweiterter ASCII-Zeichensatz verwendet, wobei die Zeichen mit den Codes 128 bis 255 für länderspezifische Sonderzeichen, Pseudografikzeichen, mathematische Symbole u. ä. verwendet werden. Auch für die Zeichen mit den Codes 1 bis 30 existieren Darstellungen auf dem Bildschirm (Symbole für Spiele usw.).

● **Alphanumerischer Monochromadapter**

Dieser Adapter kann 25 Zeilen zu 80 Zeichen darstellen. Er besitzt einen Bildwiederholspeicher von 4 KByte. Jedes Zeichen belegt 2 Bytes, den Zeichencode im IBM-Zeichensatz und ein Attributbyte. Damit kann jedes Zeichen einzeln mit den Attributen *Blinkend, Hervorgehoben, Unterstrichen* und *Invers* versehen werden. Der Bildwiederholspeicher liegt im Adreßraum des Prozessors ab 0B0000H, die Zeichen werden innerhalb einer 9 × 14-Punktmatrix mit 7 × 9 Punkten dargestellt und sind damit sehr gut lesbar.

● **CGA – Color Graphics Adapter**

Dieser Adapter besitzt einen Bildwiederholspeicher von 32 KByte, der im Adreßraum ab 0B8000H liegt. Als Controller wird (wie auch beim Monochromadapter) der Typ 6845 von Motorola eingesetzt. Der Adapter kann in 7 verschiedenen Modi betrieben werden, 4 Text- und 3 Grafikmodi /7/. Zwei Textmodi und ein Grafikmodus sind dabei farbunterdrückende Modi, die bei Anschluß eines Schwarz-Weiß-Monitors benutzt werden können. Die restlichen Modi sind /7/:

Modus	*Auflösung*	*Farben*
Text	*40 × 25 Zeichen*	*16 Vordergrund* *8 Hintergrund*
Text	*80 × 25 Zeichen*	*16 Vordergrund* *8 Hintergrund*
Grafik	*320 × 200 Pixel*	*4*
Grafik	*640 × 200 Pixel*	*2*

Textzeichen werden in einer 8 × 8-Matrix mit 5 × 7 Punkten dargestellt. Aufgrund dieser geringen Auflösung ist der CGA etwas gewöhnungsbedürftig, zumal bei der gewöhnlichen Arbeitsentfernung zum Bildschirm die Punktstruktur der Lochmaskenbildröhre sichtbar bleibt. Die Textauflösung 40 × 25 entspricht der Grafikauflösung mit 320 × 200 Pixel.
Auch hier belegt jedes Bildschirmzeichen 2 Bytes im Speicher (Zeichen und Attributbyte mit Vordergrund-/Hintergrundfarbe sowie Blinkbit). Deshalb können im Bildwiederholspeicher der CGA-Karte bis zu vier Textbildschirme gleichzeitig gehalten werden, zwischen denen umgeschaltet werden kann.
Bei der 320 × 200-Pixel-Grafik kann eine aus zwei vorgegebenen Farbpaletten gewählt werden.
Bei der 640 × 200-Pixel-Grafik kann jeder Bildpunkt entweder rückgesetzt (schwarz) oder gesetzt (eine beliebige Farbe) sein.

● **Weitere Bildschirmadapter**

Es gibt eine Reihe von weiteren Bildschirmadaptern, die breite Verbreitung gefunden haben.
Die *Hercules-Grafik-Karte* ist eine monochrome Karte mit einer Grafikauflösung von 720 × 348 Pixel. Im Textmodus entspricht die Auflösung der des Monochromadapters. Neuerdings existiert auch eine entsprechende Farbgrafikkarte /8/.
Die *IBM-EGA* (Enhanced Graphics Adapter) hat gegenüber der CGA vier weitere Modi, darunter 640 × 350 Pixel bei 64 Farben /7/. Es ist ein spezieller Farbmonitor erforderlich. Darüber hinaus existieren weitere hochauflösende Grafikadapter für Spezialzwecke.
Weiterhin gibt es eine Vielzahl sogenannter Multifunktionsgrafikkarten /8/, die hard- oder auch softwaremäßig an den erforderlichen Grafikmodus (CGA, EGA, Hercules...) angepaßt werden können.

Tastatur

Die Tastatur des IBM PC besteht aus einem alphanumerischen Tastenfeld, einem Kursortastenfeld, welches auch als Ziffernblock verwendet werden kann (durch die Taste NumLock umschaltbar), einem Funktionstastenfeld und einer Reihe von Steuertasten. Neben von CP/M-Rechnern bekannten Tasten *Shift* und *Ctrl* existiert noch eine weitere Taste zur Mehrfachbelegung der Zeichentasten, die Taste *Alt*. Damit kann jede Taste bis zu 8fach belegt werden. Der Funktionstastenblock ist bei den XT-Rechnern an der linken Seite der Tastatur angeordnet, bei den ATs und einigen kompatiblen XTs befindet er sich oberhalb des alphanumerischen Tastenfeldes. Einige At-Tastaturen haben getrennte Kursor- und Ziffernfelder und auch eine größere Anzahl Funktionstasten. Beide Rechnertypen werden mit länderspezifischen Tastaturen geliefert, deutsche Tastaturen enthalten die eckigen Klammern als Drittbelegung der Tasten ü/Ü und +/* (mit der Tastenkombination Ctrl-Alt), aber z. B. keine geschweiften Klammern! Durch Drücken der Alt-Taste und gleichzeitiges Eingeben der dezimalen Nummer des entsprechenden Zeichens auf dem Zifferntastenfeld lassen sich jedoch alle 255 Zeichen des Zeichensatzes eingeben.
Die Tastaturen liefern keinen ASCII-Code, sondern jede Taste meldet ihre Betätigung (sowohl Drücken als auch Loslassen) durch ihre Nummer, den sogenannten Scan-Code. Deshalb ist es z. B. möglich, zu erkennen, ob die rechte oder die linke oder auch beide Shift-Tasten gedrückt sind.

Speicheraufteilung

Der vom Prozessor adressierte Speicherbereich ist bei den IBM PC und Kompatiblen folgendermaßen eingeteilt /7/:

00000 RAM
A0000 Bildschirmerweiterung EGA
B0000 Bildwiederholspeicher Monochrom
B8000 Bildwiederholspeicher CGA
C0000 BIOS-Erweiterungen
F0000 ROM-BIOS

Daraus ergeben sich die maximal 640 KByte RAM, mit denen unter MS-DOS gewöhnlich gearbeitet werden kann. Die BIOS-Erweiterungen befinden sich in ROMs auf den entsprechenden Erweiterungssteckeinheiten, z. B. die Treiber für die Festplatte.

Hardwarekompatibilität

Was muß man von einem Rechner verlangen, damit er als *IBM-kompatibel* bezeichnet werden kann? Eine große Anzahl Programme für den IBM PC benutzen die Hardware des PC, ohne dafür Betriebssystemrufe zu verwenden, z. B. durch direktes Schreiben in den Bildwiederholspeicher oder gar durch Manipulation des Grafikcontrollers 6845 o. ä. Damit solche Programme auf einem anderen Rechner laufen können, müssen natürlich alle E/A-Adressen mit denen des Originals übereinstimmen. In /9/ werden zwei Arten der Kompatibilität unterschieden:
– Kompatibilität auf Bausteinebene
– funktionelle Kompatibilität.
Die erste Art liegt vor, wenn alle wesentlichen Peripheriebausteine (USART, Controller für Disketten, Grafikbildschirm usw.) mit denen des Oiginalproduktes typgleich sind. Unter diesen Umständen werden auch jene Programme, die direkt auf die Hardware zugreifen, lauffähig sein. Werden andere Bausteine benutzt, so ist diese Funktionsfähigkeit nicht mehr gegeben. Vielfach wird die Funktion des Originals dann emuliert, um eine funktionelle Kompatibilität zu erreichen. Ein Zugriff auf die entsprechenden E/A-Adressen wird durch eine spezielle Hardware erkannt und funktionell durch ein entsprechendes Programm auf den eingebauten Baustein umgesetzt. Solche Prozeduren verlangsamen natürlich den zeitlichen Ablauf, wodurch es bei zeitkritischen Anwendungen zu Problemen kommen kann. Zur praktischen Überprüfung der Kompatibilität werden häufig einige Programme empfohlen, die vom direkten Zugriff auf die Hardware ausgiebig Gebrauch machen und deren Lauffähigkeit deshalb als Indikator für IBM-Kompatibilität gelten kann. Ein solches Programm ist z. B. der Flugsimulator von Microsoft /9/.

Das Betriebssystem

● **Geschichtlicher Überblick /10/**

Als Betriebssystem für den PC wählte IBM 1981 das Betriebssystem MS-DOS der Firma Microsoft. Für den IBM PC bekam es die Bezeichnung PC-DOS. PC-DOS wurde im wesentlichen in Zusammenarbeit mit Microsoft weiterentwickelt, so daß beide Systeme auch in den jüngsten Versionen nur minimal voneinander abweichen. Grundsätzlich läuft PC-DOS nur auf original IBM-Geräten. Alle Kompatiblen verwenden MS-DOS. In der Version 1.0 war MS-DOS sehr an das Standardbetriebssystem für 8-Bit-Rechner mit dem Prozessor 8080, CP/M, angelehnt. Da 8080-Programme recht einfach auf den 8086/8088 übertragen werden können, gelang es damit, viele der von CP/M her bekannten Programmsysteme schnell unter MS-DOS verfügbar zu machen. Das betrifft zum Beispiel die bekannten Programme WordStar, dBase II, Multiplan. Bis zum Jahre 1983 wurde MS-DOS erheblich weiter entwickelt. Die Version 2.0 von MS-DOS war eigentlich ein komplett neues Betriebssystem, wobei die volle Kompatibilität zur Version 1 erhalten blieb. Die Version 2.0 unterstützte doppelseitige Disketten mit 9 Sektoren pro Spur (vorher nur 8 Sektoren) und Festplattenlaufwerke. Daneben wurde eine Vielzahl von neuen, durch das Betriebssystem UNIX inspirierten Features in MS-DOS aufgenommen. Die Version 2.11 wird noch heute teilweise von Herstellern kompatibler Rechner geliefert. Mit dem IBM PC/AT erschien 1984 die MS-DOS-Version 3.0 und kurz danach 3.1. Neben der Unterstützung größerer Festplatten wurde MS-DOS netzwerkfähig. Aus dem Jahre 1986 ist die Version 3.2. Hier wurde besonders die Anpassung anderer Medien, z. B. von 80-Spur-Diskettenlaufwerken, erleichtert.
Inzwischen gibt es eine Version 3.3, die die Unterstützung von Festplatten größer 32 MByte durch Partitionierung ermöglicht.

● **Die Bestandteile von MS-DOS**

MS-DOS besteht aus den drei Teilen BIOS (Basic I/O System), dem DOS (Disk Operating System) und dem Kommandoprozessor. Das BIOS ist ähnlich wie bei CP/M der hardwareabhängige Teil des Betriebssystems MS-DOS. Es enthält die Treiber für die Standardperipheriegeräte und ist auf der MS-DOS-Systemdiskette als physisch erstes File unter dem Namen IO.SYS oder IO.BIN (bei PC-DOS unter IBMBIO.COM) enthalten. Beim IBM PC nutzt das BIOS die Treiber des

ROM-BIOS, bei Kompatiblen werden die Funktionen des ROM-BIOS des Original-PC vollständig oder teilweise vom RAM-BIOS übernommen.
Der zweite Teil des Betriebssystems ist der Kern des DOS.
Auf der Systemdiskette ist er das physisch zweite File und hat den Namen MSDOS.SYS oder IBMDOS.COM. Das DOS stellt den Nutzerprogrammen eine Vielzahl von Systemaufrufen für
– die Verwaltung von Dateien
– die Verwaltung des Hauptspeichers
– Eingabe von und Ausgabe nach der zeichenorientierten Peripherie
– Zugriff zur eingebauten Uhr
zur Verfügung. Alle Systemdienste werden über sogenannte Softwareinterrupts aufgerufen. Auch die Dienste des ROM-BIOS benutzen diese Aufrufart.
Der dritte Teil des Betriebssystems ist der Kommandoprozessor COMMAND.COM. Er übernimmt (ähnlich dem CCP von CP/M) die Kommunikation mit dem Nutzer des Rechners. Der Kommandoprozessor besteht aus einem residenten und einem transienten Teil. Der transiente Teil kann durch Anwenderprogramme überschrieben werden und wird dann vom residenten Teil nachgeladen. Es besteht die Möglichkeit, COMMAND.COM durch einen eigenen, anwenderspezifischen Kommandoprozessor zu ersetzen.

Literatur

/1/ Neues vom PC. Mikroprozessortechnik, Berlin, 1 (1987) 8, S. 243
/2/ Ein PC für alle Fälle. PC-Magazin (1987) 19, S. 44–48
/3/ Vieillefond, C.: Programmierung des 80286. Sybex-Verlag, Düsseldorf 1987
/4/ Grafik, W.: Aufbau und Wirkungsweise von 16-Bit-Mikroprozessoren. RA 226, VEB Verlag Technik, Berlin 1987
/5/ Koeppe, P.: Tips zur Hardware-Konfiguration. PC-Magazin (1987) 28, S. 64–68
/6/ Marktübersicht Systeme mit 80286-Prozessor. PC-Magazin (1987) 35, S. 19–25
/7/ Norton, P.: Programmierhandbuch für den IBM PC. Vieweg-Verlag, Braunschweig 1986
/8/ Marktübersicht Grafikplatinen. PC-Magazin (1987) 31, S. 24–25
/9/ Smode, D.: Das große MS-DOS-Profi-Arbeitsbuch. Franzis-Verlag, München 1987

KONTAKT

Ingenieurhochschule Mittweida, Sektion Informationselektronik, Platz der DSF 17, Mittweida, 9250; Tel. 58346

Wegbereiter der Informatik



GOTTFRIED WILHELM von LEIBNIZ

* 1646 Leipzig † 1716 Hannover

Leibniz gehört unangefochten zu den vielseitigsten und geistreichsten Universalgelehrten, die die Menschheit hervorgebracht hat. Der englische Mathematiker Bertrand Russel zählte ihn zu den „größten Denkern aller Zeiten".
In der Tat war Leibniz nicht nur ein bedeutender Mathematiker und Logiker, sondern ebenso Naturforscher, Erfinder, Rechtsgelehrter, Philosoph, Politiker, Geschichts- und Sprachforscher. Sein scharf denkender Geist befähigte ihn als achtjährigen zum Beispiel, quasi mittels logischer Dechiffrierung eines illustrierten, lateinischen Textes ohne Anleitung Latein zu lernen und danach in dieser Sprache mit großer Schnelligkeit Gedichte zu verfassen.
Er führte einen erstaunlich umfangreichen Briefwechsel mit der gesamten wissenschaftlichen und politischen Welt seiner Zeit – es wird bis ins nächste Jahrhundert dauern, bis sein gesamter schriftlicher Nachlaß durchdacht und veröffentlicht sein wird. Er ist der Gründer der Berliner Akademie der Wissenschaften (1700) und war ihr erster Präsident, er regte (1711) auch die Gründung der Petersburger Akademie an und war Mitglied der Londoner Royal Society sowie der Akademie Paris. Er arbeitete unablässig, mit 5 Stunden Schlaf täglich kam er aus. Zeitweilig muß er ein Sklave seiner Ideen gewesen sein – so notierte er über sich selbst: „Beim Erwachen hatte ich schon so viele Einfälle, daß der Tag nicht ausreichte, um sie niederzuschreiben".
So entwarf er Pläne für den Bau eines Unterseebootes, erfand das Anemometer, konstruierte einen Windmotor (für Wasserpumpen); er entwickelte Geräte, die das Grubenwasser in Bergwerken absaugten, erfand die Staffelwalze für den Zehnerübertrag in seiner mechanischen Rechenmaschine; er behandelte das Brechungsgesetz der Optik als Extremalproblem, analysierte die Fallbewegung eines Körpers im zähen Medium; er verteidigte den Evolutionsgedanken in der Biologie und erbrachte (200 Jahre vor S. Freud) den logischen Nachweis, daß der Mensch neben seinem Bewußtsein auch ein Unterbewußtsein haben müsse.
Keinem Geringeren als G. E. Lessing, der sich intensiv mit der Leibnizschen Philosophie beschäftigt hat, verdanken wir eine der ersten chronologischen Kurzbiographien über Leibniz. Die wesentlichsten Lebensdaten seien kurz angegeben: Als Sohn eines Juristen und Professors für Moralphilosophie trat Leibniz mit 15 Jahren in die Leipziger Universität ein, um Jura und (teilweise auch in Jena) nebenher Philosophie, Logik und Mathematik zu studieren. Im Jahre 1664 erwarb er bereits die Magisterwürde und 1666 wurde er Doctor Juris in der später erloschenen Universität Altdorf bei Nürnberg (die Leipziger Universität hatte ihn wegen seines zu jungen Alters abgewiesen). 1670 trat er für 6 Jahre als Hofrat in den Dienst des Mainzer Kurfürsten; 1672 weilte er in diplomatischer Mission in Paris, wo er die Arbeiten von Galilei, Cartesius, Fermat, Pascal und Huygens studierte. In Paris arbeitete er auch wesentliche Fragen der Differentialrechnung aus. Im Jahre 1676 trat er in die Dienste des Herzogs von Braunschweig-Lüneburg, der ihn als Hofrat und Oberbibliothekar nach Hannover berief; in diesem Amt blieb er 40 Jahre lang bis zu seinem Tode. Auf Grund seiner herausragenden Verdienste wurde er 1707 von Kaiser Karl VI. zum Freiherrn geadelt und zum Reichshofrat ernannt.
Besonders schöpferisch war Leibniz auf dem Gebiet der Mathematik tätig. Da sei zunächst die von ihm konstruierte Rechenmaschine genannt, die er 1673 in der Royal Society in London vorführte. Es war die erste Maschine, die außer addieren und subtrahieren auch noch multiplizieren, dividieren, potenzieren sowie die 2. und 3. Wurzel ziehen konnte. Leibniz war übrigens auch der erste, der Pläne für eine Rechenmaschine entwarf, welche auf dem von ihm entwickelten binären Zahlensystem basierte! Einer solch kühnen, der Zukunft weit vorgreifenden Vorstellung konnte die Technik der Barockzeit allerdings noch nicht folgen. Erst in der von Konrad Zuse 1941 konstruierten, ersten programmgesteuerten Rechenanlage der Welt wurde das binäre Zahlensystem dann technisch realisiert.
Leibniz legte auch die Grundlagen zur formalen Logik. Des weiteren untersuchte er die Entwicklung von Funktionen in eine Reihe und arbeitete das nach ihm benannte Konvergenzkriterium aus, er definierte den Begriff der Determinante und verfaßte die Grundlagen der Determinantentheorie, die dann von Vandermonde und Gauß weiterentwickelt und von Jacobi faktisch abgeschlossen wurde.
Sein wichtigstes mathematisches Verdienst ist jedoch die Erfindung der Differential- und Integralrechnung. Dabei ging er nicht – wie Newton – von der Quadratur, sondern von der Fragestellung nach der Tangente aus; anschließend lieferte er den Beweis, daß aus dem „Umkehrproblem der Tangentenbildung die Quadratur aller Figuren herleitbar" ist. Er leitete die Integrationsgesetze ab sowie die Rechenregeln für die Differentiation eines Produktes, einer Potenz und für die Differentiation von impliziten Funktionen (Veröffentlichung 1684). Er machte sogar den Versuch, die Differentiation $\frac{d^n y(x)}{dx^n}$ für beliebige reelle n zu erweitern!
Im Jahre 1693 veröffentlichte Leibniz erstmals Methoden zur Lösung von Differentialgleichungen mit Hilfe unendlicher Reihen. Leibniz gehört neben Cartesius und Euler zu den Hauptgestaltern der heutigen mathematischen Formelschreibweise. So führte er die Indizes ein, die Potenzschreibweise a^x, die Determinantenschreibweise, logische Symbole, die Differentiations- und Integrationszeichen, des weiteren die Termini: Funktion, Abszisse, Ordinate, Koordinate, Differentialgleichung, Algorithmus u. v. a. m.
Leibniz durchbrach übrigens als erster die jahrhundertealte Tradition, wissenschaftliche Werke nur in lateinischer Sprache zu veröffentlichen. Erwähnt seien noch seine Entwürfe einer Universalsprache und einer Universalschrift, die ihn sein ganzes Leben hindurch beschäftigten.
Seine bis heute aktuelle Forderung THEORIA CUM PRAXI hat er selbst in hervorragender Weise in die Tat umgesetzt.

Dr. Klaus Biener

Mikroprozessorsystem K 1810 WM86

Hardware · Software · Applikation (Teil 3)

Prof. Dr. Bernd-Georg Münzer
(wissenschaftliche Leitung),
Dr. Günter Jorke, Eckhard Engemann,
Wolfgang Kabatzke, Frank Kamrad,
Helfried Schumacher, Tomasz Stachowiak
Wilhelm-Pieck-Universität Rostock,
Sektion Technische Elektronik,
Wissenschaftsbereich Mikrorechentechnik/
Schaltungstechnik

Tafel 4.1 Initialisierungskommandos

ICW 1 — *Portadresse mit A0 – 0*

D7	D6	D5	D4	D3	D2	D1	D0
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	SNGL	IC4

LTIM	*Eingangssteuerung für IR-Eingänge*
1	*pegelgesteuert ('1'-aktiv)* (level triggered mode)
0	*flankengesteuert ('01'-Flanke)* (edge triggered mode)
SNGL	*Einzel- oder Multi-PIC-System*
0	*kaskadierte PICs, ICW3 erforderlich*
1	*Einzel-PIC, ICW3 nicht erforderlich*
IC4	*Ausgabe von ICW4*
0	*nicht erforderlich*
1	*erforderlich*

ICW2 — *Portadresse mit A0 = 1*

D7	D6	D5	D4	D3	D2	D1	D0
T7	T6	T5	T4	T3	x	x	x

T7–T3	*höherwertiger 5-Bitanteil des Interruptcode*

ICW3 — *Portadresse mit A0 = 1*

Mastermode:

D7	D6	D5	D4	D3	D2	D1	D0
S7	S6	S5	S4	S3	S2	S1	S0

S7–S0	*Slave-PIC-Anschluß*
0	*einfache Interruptquelle*
1	*an IR-Eingang ist ein Slave-PIC angeschlossen*

Slavemode:

D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	x	ID2	ID1	ID0

ID2–ID0	*Identifikationscode für Slave-PIC*

ICW4 — *Portadresse mit A0 = 1*

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	SFNM	BUF	M/S	AEOI	1

AEOI	*Interruptfreigabeform*
0	*Rücksetzen des ISR-Bits mit EOI-Kommando (Normal EOI)*
1	*automatisches Rücksetzen der höchstpriorisierten ISR-Bitstelle nach Interruptannahme (nur im Mastermode)*
BUF	*Puffer-Mode*
0	*keine Puffersteuerung*
1	*Steuerausgang für Daten-BUS-Treiber auf SP / EN*
M/S	*PIC-Arbeitsweise im Puffer-Mode (BUF – 1)*
0	*Slave-Mode*
1	*Master-Mode*
SFNM	*Interruptverschachtelungsmode*
0	*keine Interruptverschachtelung für Anforderungen innerhalb eines Slave-PIC*
1	*volle Interruptverschachtelung bei kaskadierten PICs, (special full nested mode)*

Tafel 4.1 enthält die Initialisierungskommandos **ICW1** bis **ICW4**.
Im **AEOI**-Mode (automatic end of interrupt) wird die der laufenden Interrupt-Service-Routine zugeordnete Bitstelle im ISR bereits mit der fallenden Flanke des zweiten **INTA**-Impulses zurückgesetzt. Dadurch kann auf das Rücksetzen am Ende des Interruptprogramms mit einem speziellen **PIC**-Kommando verzichtet werden. Innerhalb einer prioritätsgesteuerten verschachtelten Interruptstruktur ist dieser Mode jedoch nicht sinnvoll.
Nach der Einstellung des Puffermodes erzeugt der **PIC** am Ausgang **SP/EN** für die Dauer der Ausgabe des Interruptvektors ein Steuersignal zur Ansteuerung der Daten-**BUS**-Puffer. Im Puffermode legt die Bitstelle **M/S** die Reaktion als Master- oder Slave-**PIC** fest.
Mit **Operationssteuerkommandos** (operation control word, OCW) lassen sich weitere Betriebsarten des PIC jederzeit während der Programmabarbeitung einstellen (Tafel 4.2). Operationssteuerworte können in beliebiger Reihenfolge ausgegeben werden.

Tafel 4.2 Operationskommandos

OCW1 — *Portadresse mit A0 = 1*

D7	D6	D5	D4	D3	D2	D1	D0
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

M7–M0	*Interrupteingangsmaskierung*
0	*Interrupteingang freigegeben*
1	*Interrupteingang gesperrt*

OCW2 — *Portadresse A0 – 0*

D7	D6	D5	D4	D3	D2	D1	D0
R	SL	EOI	0	0	L2	L1	L0

L7–L0	*Binärcode für Interrupteingangsnummer*
R SL EOI	*Kommando*
0 0 1	*EOI-Kommando (end of interrupt) mit L0, L1, L2 – 0*
0 1 1	*SEOI-Kommando (special end of interrupt)*
1 0 1	*Rotation im EOI-Mode mit L0, L1, L2 = 0*
1 0 0	*Rotation im AEOI-Mode, Setzen mit L0, L1, L2, – 0*
0 0 0	*Rotation im AEOI-Mode, Löschen mit L0, L1, L2 = 0*
1 1 1	*Rotation im SEOI-Mode*
1 1 0	*Priorität setzen*
0 1 0	*keine Operation*

OCW3 — *Portadresse mit A0 = 0*

D7	D6	D5	D4	D3	D2	D1	D0
x	ESMM	SMM	0	1	P	RR	RIS

RR RIS	*Statuslesekommando mit D2, D5, D6 – 0*
1 0	*nächster Lesebefehl bezieht sich auf IRR*
1 1	*nächster Lesebefehl bezieht sich auf ISR*
0 x	*keine Leseoperation*
P	*Pollingkommando mit D0, D1, D5, D6 – 0*
1	*eingeschaltet*
0	*ausgeschaltet*
ESMM SMM	*spezieller Maskierungsmode mit D0, D1, D5, D6 = 0*
1 1	*speziellen Maskierungsmode einstellen*
1 0	*speziellen Maskierungsmode abstellen*
0 x	*keine Maskierungsmodeeinstellung*

Das Operationssteuerwort **OCW1** setzt das Interruptmaskenregister. Auch für gesperrte Interrupteingänge werden jeoch bei flankengesteuerten Eingängen Interruptanforderungen im IRR abgespeichert.
Mit dem Operationssteuerwort **OCW2** werden sieben Kommandos der Interruptfreigabe und der Interruptorganisation gebildet: Die Ausgabe eines **EOI**-*Kommandos* (end of interrupt) setzt die höchstpriorisierte aktive Bitstelle des **ISR** zurück.
Das **SEOI**-*Kommando* (specific end of interrupt) bezieht sich auf eine ausgewählte Bitstelle des **ISR**.
Nach der Initialisierung sind die Prioritäten den **IR**-Eingängen fest zugeordnet, wobei der Eingang **IR0** die höchste Priorität besitzt. Diese Prioritätsorganisation kann durch die Rotationskommandos abgeändert werden. Durch die Rotation der Prioritäten erhalten alle **IR**-Eingänge die gleichen Zugriffsmöglichkeiten. Die Interruptrotation ist den verschiedenen Formen der Interruptfreigabe des **IRR** (**EOI**-, **SEOI**- und **AEOI**-Mode) in unterschiedlichen Kommandos zugeordnet.
Das Kommando *Rotation im* **EOI**-*Mode* veranlaßt nach dem Rücksetzen der höchstpriorisierten Bitstelle des **ISR** die Zuordnung der niedrigsten Priorität (Wert 7) für den zugehörigen **IR**-Eingang. Die diesem Eingang folgenden Eingänge erhalten, beginnend mit Priorität 0, abnehmende Prioritäten zugewiesen. Auf diese Weise wird gesichert, daß eine erneute Interruptanforderung für einen **IR**-Eingang erst wieder berücksichtigt wird, wenn alle anderen anfordernden Eingänge bedient worden sind. Die höchste Interruptpriorität rotiert nach jeder Interruptbearbeitung zum nächsthöheren **IR**-Eingang.
Die Prioritätsrotation erfolgt im **AEOI**-Mode automatisch nach dem zweiten **INTA**-Impuls bei der Interruptannahme. Mit den Kommandos *Rotation im* **AEOI**-*Mode; Setzen* und *Rotation im* **AEOI**-*Mode; Löschen* wird die Rotation im AEOI-Mode ein- und ausgeschaltet.
Die niedrigste Priorität kann einem **IR**-Eingang mit dem Kommando *Priorität setzen* zugewiesen werden. Damit wird zugleich für alle folgenden Eingänge, beginnend mit der höchsten Priorität, eine abnehmende Priorität eingestellt. Diese Neueinstellung wird erst für die danach eintreffenden Interruptanforderungen wirksam.
Das Kommando *Rotation im* **SEOI**-*Mode* kombiniert die Funktionen der Kommandos für das Einstellen des **SEOI**-Modes und des Interruptsetzens. Die ausgewählte Bitstelle des **IRR** wird gelöscht und dem zugehörigen Eingang die niedrigste Priorität zugewiesen. Alle folgenden **IR**-Eingänge erhalten neue Prioritäten in der o. a. Form.

Leseoperationen:
Die Register **IRR**, **ISR** und **IMR** können gelesen werden. Vor einer Leseoperation für die Portadresse mit A0 = 0 muß der Zugriff auf **IRR** oder **ISR** durch eine Einstellung mit dem Kommandosteuerwort **OCW3** ausgewählt

werden. Diese Auswahl bleibt bis zur Neueinstellung auch über mehrere Leseoperationen erhalten. Nach der Initialisierung wird der Zugriff auf **IRR** eingestellt.

Um das Maskenregister IMR zu lesen, ist eine Voreinstellung nicht notwendig, da dafür die Portadresse mit **A0** = 1 benutzt wird.

Durch **OCW3** kann auch ein *spezieller Maskierungsmode* eingestellt werden. In diesem Mode werden Interruptbearbeitungen für Anforderungen mit einer Priorität, die geringer ist als die der in der Abarbeitung befindlichen Interruptoutine, ermöglicht. Weitere Anforderungen des bearbeiteten **IR**-Eingangs werden jedoch nicht bedient, um eine unkontrollierte Verschachtelung desselben Interruptprogramms zu vermeiden.

Tafel 4.3 Programmierbeispiel

```
CODSEG        EQU  1000H
INTOFF        EQU  0E8H           ;Adresse in der
                                  ;Interrupttabelle
;
PIC0          EQU  0C0H           ;PIC-Adresse mit A0=0
PIC1          EQU  0C2H           ;PIC-Adresse mit A0=1
PICICW1       EQU  17H            ;Einzel-PIC,
                                  ;flankengesteuert
PICICW2       EQU  38H            ;Interruptvektor
PICICW4       EQU  0FH            ;Master- und Buffer-
                                  ;mode, AEOI
PICOCW1       EQU  0FBH           ;Interruptfreigabe
                                  ;für IR2
;
TIMCH0        EQU  0D0H           ;Timeradresse für
                                  ;Kanal 0
TIMCON        EQU  0D6H           ;Timeradresse für
                                  ;Steuerkanal
TIMMD0        EQU  36H            ;Steuerwort Timer
                                  ;(Mode 3, dual)
TIMOL         EQU  02H            ;Zeitkonstante,
                                  ;unterer Anteil
TIMOH         EQU  30H            ;Zeitkonstante,
                                  ;oberer Anteil
USART_CONTROL      EQU   0DAH     ;Portadresse USART,
                                  ;Control
USART_DATA    EQU  0D8H           ;Portadresse USART,
                                  ;Daten
              DSEG 0              ;Adreßeinstellung in
              ORG  INTOFF         ;Interruptadreßtabelle
              DW   OFFSET ISR     ;Adresse der Inter-
              DW   CODSEG         ;rupt-Service-Routine

              CSEG CODSEG
TEST:         CLI                 ;CPU-Interrupt sperren
              MOV  AL,PICICW1     ;PIC-Initialisierungs-
                                  ;kommandos
              OUT  PIC0,AL        ;ausgeben
              MOV  AL,PICICW2
              OUT  PIC1,AL
              MOV  AL,PICICW4
              OUT  PIC1,AL
              MOV  AL,PICOCW1     ;PIC-Interruptmaske
                                  ;ausgeben
              OUT  PIC1,AL
              MOV  AL,TIMMD0
              OUT  TIMCON,AL      ;Mode-Einstellung
                                  ;Timer
              MOV  AL,TIMOL       ;Ausgabe der Zeitkon-
                                  ;stante
              OUT  TIMCH0,AL      ;für Timer
              MOV  AL,TIMOH
              OUT  TIMCH0,AL
              STI                 ;CPU-Interrupt frei-
                                  ;geben
LP:           JMPS LP             ;endlose Schleife
;
ISR:          MOV  AL,'*'
              CALL OUTPUT         ;Bildschirmausgabe
              IRET
;
OUTPUT:       PUSH AX
OUTPUT1:      IN   AL,USART_CONTROL
                                  ;Lesen Statusregister
              TEST AL,1           ;Prüfen TxRDY = 1
              JZ   OUTPUT1
              POP  AX             ;Sendepuffer leer!
              OUT  USART_DATA,AL  ;Daten schreiben
              RET
```

Der **PIC** kann auch für die Unterstützung von Pollingverfahren eingesetzt werden. Nach der Ausgabe des *Polling-Kommandos* kann mit einem einzigen Lesebefehl festgestellt werden, ob an einem **IR**-Eingang des PIC eine Bedienanforderung anliegt. Dieser Fall wird mit **D7** = 1 des eingelesenen Bytes gekennzeichnet. Die drei unteren Bits dieses Bytes enthalten die IR-Eingangsnummer der höchstpriorisierten Anforderung.

4.5 Programmbeispiel

Das Programmbeispiel in Tafel 4.3 behandelt die Anschaltung eines Programmierbaren Intervall-Timers **8253** an den **PIC.** Dabei wird der Ausgang des Zählers 0 mit dem Eingang **IR2** des **PIC** verbunden.

Der Timer erzeugt im Mode 3 eine Impulsfolge im Zeitabstand von 10 ms (s. Abschn. 3). Die **PIC**-Progammierung enthält innerhalb eines Einzel-**PIC**-Systems die Interruptfreigabe für den Eingang **IR2** mit Flankensteuerung. Aus der Angabe des Interruptvektors **38H** ergibt sich für den Eingang **IR2** die Adresse **0:0E8H** innerhalb der Interruptadreßtabelle. Auf dieser Speicherposition wird die Adresse des Interrtupt-Service-Programms angegeben.

Nach der Programmierung des Timers und des **PIC** verbleibt das Hauptprogramm in einer endlosen Schleife. Im Abstand von 10 ms wird die Interrupt-Service-Routine eingeblendet, in der das Zeichen '*' auf dem Bildschirm ausgegeben wird. Die Programmformulierung enthält die mnemonischen Befehlsbeschreibungen und Pseudoanweisungen des Assembler-Programmes **ASM86**, die im Abschnitt 5 dieser Folge beschrieben werden.

5. Assemblerbefehle der 8086-CPU

5.1 Befehlsübersicht

Der Befehlssatz der **8086-CPU** enthält die wichtigsten, aus der 8-Bit-Technik bekannten Funktionsgruppen, welche jedoch durch eine Reihe weiterer effektiver Befehle ergänzt wurden.

Die CPU realisiert die Verarbeitung von 8-Bit-Daten (Bytes) und 16-Bit-Daten (Wörter).

Arithmetische Operationen beinhalten verschiedene Formen der Addition und Subtraktion, der Multiplikation und der Division.

Logische Operationen beziehen sich auf **UND**-, **ODER**- und **XOR**-Verknüpfungen zwischen zwei Operanden und die Negation, Rotation und Verschiebung von Einzeloperanden.

Stringoperationen gestatten das Umladen und den Vergleich von Speicherblöcken und die Abspeicherung bzw. Suche von Zeichen in Speicherbereichen.

Für die Programmablauforganisation steht eine Vielzahl unbedingter und bedingter Sprünge, Schleifenanweisungen, direkte und indirekte Unterprogrammaufrufe und Returnanweisungen zur Verfügung.

Die Registerstruktur des 8086 zeigen die Bilder 2.3, 2.4 und 2.5.

Die vier Hauptregister **AX**, **BX**, **CX** und **DX** können als 16-Bit-Register oder 8-Bit-Register mit den Bezeichnungen **AL**, **AH**, **BL**, **BH**, **CL**, **CH**, **DL** und **DH** benutzt werden. Die Indexregister **SI** (source index register) und **DI** (destination index register) und das Pointerregister **BP** (base pointer) werden als 16-Bit-Operandenregister oder für die indirekte Speicher-Adressierung vewendet. Der Stackpointer **SP** verwaltet den Prozessorstack.

Die Hauptregister, Indexregister und Pointerregister können in der Mehrzahl aller Befehle als Operandenregister verwendet werden.

5.2 Assemblerprogrammierung

Im folgenden werden alle Befehle auf dem Niveau der Assemblerbeschreibung erläutert, wobei auf das Assemblerprogramm **ASM86** im Betriebssystem **SCP 1700** Bezug genommen wird, das mit dem gleichnamigen Cross-Assembler im Betriebssystem **SCPX** übereinstimmt.

Auf Besonderheiten des Assemblerprogrammes **RASM86** im Betriebssystem **SCP 1700** wird im Abschnitt 6 hingewiesen.

Die Beschreibung der Befehle erfolgt in mnemonischer Darstellung, wobei die Operanden in Registern mit dem Registernamen dargestellt sind.

Konstanten können in dualen, oktalen, dezimalen und hexadezimalen Darstellungen oder im **ASCII**-Code auftreten.

5.2.1 Speicheradressierung

Bei der *direkten* Adressierung von Speicherplätzen werden konstante Adressen numerisch oder als Symbol angegeben (Bild 5.1.a).

OFFSET = 16 Bit offset (im Befehl).

Bei der *indirekten* Adressierung ist

- die Offsetadresse in einem der 16-Bit-Register enthalten:
  - Basisregister [**BX**], [**BP**] oder
  - Indexregister [**SI**], [**DI**].

Die Registerangabe erfolgt in eckigen Klammern (Bild 5.1.b).

OFFSET = [**BX**] oder [**BP**] oder [**SI**] oder [**DI**].

- die Offsetadresse als Summe von zwei 16-Bit-Registern angegeben:
  - Basisregister + Indexregister

OFFSET = [**BX**+**SI**] oder [**BX**+**DI**] oder [**BP**+**SI**] oder [**BP**+**DI**] (Bild 5.1.c)

Bild 5.1 Adressierungsarten des 8086
a) Direktoperand
b) Operand in Indexregister SI,DI oder Basisregister BX,BP
c) Operand in Summe von Indexregister plus Basisregister

• eine zusätzliche positive oder negative Verschiebung (displacement) wird im Assemblerprogramm **ASM86** außerhalb der eckigen Klammern ergänzt, während einige Reassembler, z. B. im Monitor des **A 7100**, dafür die sinnvollere Darstellung aller Bestandteile der Adresse innerhalb der eckigen Klammern benutzen.

Beispiel:

OFFSET = [**SI** + **DI**] + displacement

Die *Standard-Zuweisung* der Operanden zu den Segmentregistern lautet:

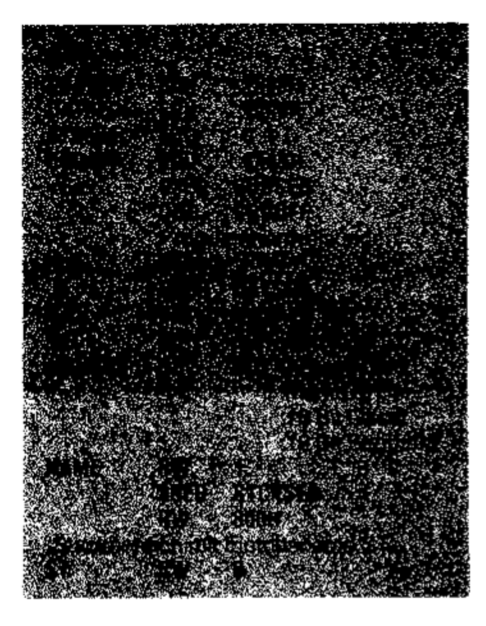
Operand	*Segmentregister*
Direktoperand, **BX**, **SI**, **DI**	Datensegment **DS**
SP, **BP**	Stacksegment **SS**
DI (Stringoperationen)	Extrasegment **ES**

Wenn ein Indexregister zusammen mit einem Basisregister benutzt wird, gilt das Standard-Segmentregister des Basisregisters als Standardsegment für den gesamten Operanden:
[**BP** + **SI**] mit Segment **SS**
[**BX** + **DI**] mit Segment **DS**

Speicheradreßangaben mit vom Standard abweichenden Segment-Registern werden durch Vorsetzen des neuen Segmentregisternamens dargestellt (Segmentpräfix).
Der ASM86 verarbeitet über den Umfang der Prozessorbefehle hinaus sogenannte *Pseudoanweisungen*, wovon nur einige wichtige hier angegeben sind.
Mit der **EQU**-Anweisung werden Symbole (Namen) für numerische Konstanten eingeführt. Diese können auch über einfache Ausdrücke definiert werden.



Mit den Pseudoanweisungen **DB** (define byte), **DW** (define word), **RB** (reserve byte) und **RW** (reserve word) können im Codesegment, Datensegment, Stacksegment und Extrasegment Speicherbereiche definiert und benannt werden. Die Anweisungen DB und DW initialisieren den Speicherbereich mit den in den anschließenden Listen byte- oder wortweise angegebenen Konstanten. Hinter RB und RW steht nur die Zahl der reservierten Bytes oder Wörter.
Die **DD**-Anweisung initialisiert vier Bytes im Speicher mit einer vollständigen Offset- und Segmentadresse.
Die Segmentzuordnung zur definierten Adresse für Programm- und Datenbereiche erfolgt im Assemblerprogramm ASM86 mit den Pseudoanweisungen **CSEG**, **DSEG**, **SSEG** und **ESEG** für das Code-, Daten-, Stack- und Extrasegment.
Der Inhalt des **CS**-Registers wird beim Programmstart eingestellt, während die restlichen Segmentregister *vor* ihrer Verwendung im Programm geladen werden müssen.
Mit der Anweisung **ORG** wird die Offsetadresse innerhalb des Segments festgelegt.



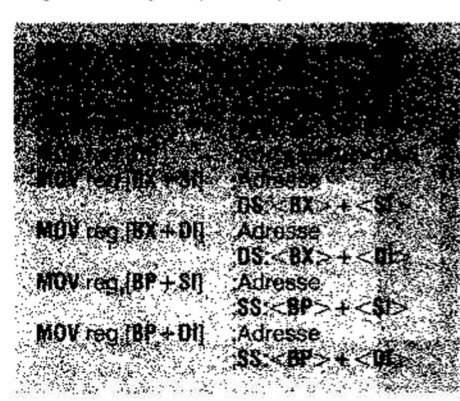
5.3 8086-Befehle

5.3.1 Transportbefehle

MOV-Befehle dienen dem Transport von 8- und 16-Bit-Daten zwischen den Registern untereinander, zwischen Registern und Speicherplätzen und von 8- und 16-Bit-Konstanten in Register und auf Speicherplätze.
Als 16-Bit-Register in **MOV**-Befehlen sind die vier Hauptregister, die Indexregister, die Pointerregister und mit Einschränkungen die Segmentregister zugelassen.
Die folgenden Beispiele vermitteln die Registerschreibweise in Register-Register-Befehlen in Assemblernotierung:



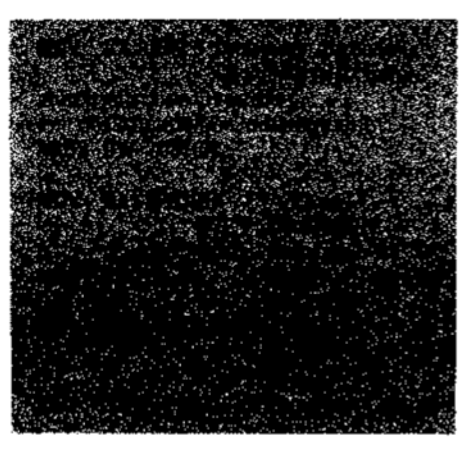
Bei MOV-Befehlen sind für den Speicherzugriff die im Abschnitt 5.2.1 genannten direkten und indirekten Adressierungsarten mit den Standard-Zuweisungen zu den Segmentregistern möglich (Bild 5.1):



Die folgenden *Beispiele* für den Transfer zwischen Registern und Speicherinhalten geben die Assemblernotierung für die verschiedenen Adressierungsformen an. Aus der Art des beteiligten Registers ergibt sich ein Byte- oder Wortzugriff auf den Speicher.



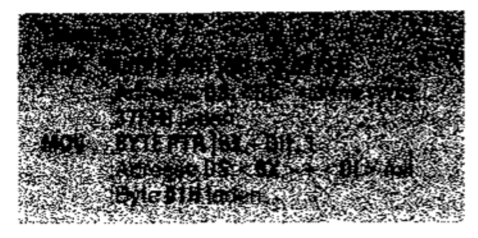
Bei Angabe von Adreß-Symbolen, die nicht im Datensegment definiert sind, erzeugt der Assembler automatisch ein entsprechendes Segment-Override-Präfix.



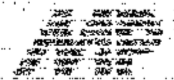
Für die Definition von im Befehl angegebenen Konstanten ist die Breite des Zielregisters entscheidend.



MOV-Befehle für das Einschreiben von *Konstanten* in den Speicher oder Befehle mit nur einem Operanden, für den auf den Speicher zugegriffen werden muß, benötigen in der Assemblernotierung die Angabe zu einem Byte- oder einem Worttransfer. Das erfolgt durch den Zusatz **BYTE PTR** oder **WORD PTR** vor der Angabe der Speicheradresse.
Diese Angaben sind auch bei einer Reihe anderer Befehle mit Konstanten-Darstellung erforderlich.

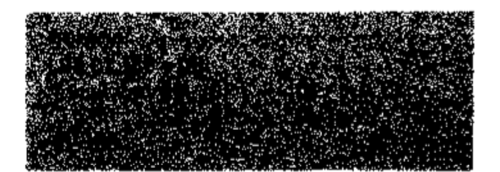


Mit Adreßtransfer-Operationen können Offsetwerte und auch Segmentwerte in Register geladen werden.
Der Befehl **LEA** (load effextive address) schreibt die Offsetadresse in eines der acht möglichen 16-Bit-Register.



Beispiel:
LEA BX,[BX+SI+0A67H]
Summe von <BX>+<SI>+0A67H in <BX> laden

Die Befehle **LDS** und **LES** übertragen den Inhalt von 2 benachbarten Speicherplätzen in ein 16-Bit-Register und den Inhalt der beiden nachfolgenden Speicherplätze in **DS** bzw. **ES**.



Der Spezialfall eines Transportbefehls **XLAT** (translate) dient zur Übertragung des Inhaltes eines Tabellenelements. Dabei wird aus der Summe der Inhalte von BX (Startadresse der Tabelle) und **Al** (displacement) eine Offsetadresse gebildet, mit der ein Byte vom Datensegment in **AL** geladen wird.



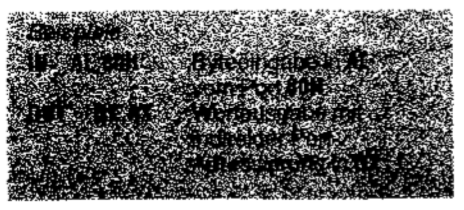
XCHG-Befehle (exchange) realisieren den Austausch von Bytes oder Wörtern zwischen Registern oder zwischen einem Rtegister und einem Speicherplatz.

Beispiel:
XCHG SI,[DI]
<SI> mit Inhalt von DS:<DI> tauschen

PUSH-und **POP**-Befehle transferieren Wortinformationen zwischen einem Register und dem Prozessorstack. Vor jedem Speicherschreiben wird die Adresse im **SP**-Register dekrementiert und nach jedem Lesen inkrementiert. Als zu transferierende 16-Bit-Register sind neben den Hauptregistern, den Indexregistern und Pointerregistern auch die Segmentregister zugelassen.
Die speziellen Befehle **PUSHF** und **POPF** beziehen sich auf die Stackoperationen für das Flagregister.

Beispiele:
PUSH DS ;DS in Stack
POP BX ;BX aus Stack
PUSHF
POPF

Die Flagladeoperation **LAHF** (load AH with flags) überträgt die dem Prozessor **8080** entsprechenden Flags des **8086** in der Anordnung **SF,ZF,x,AF,x,PF,x,CF** in das Register **AH**.
Der Befehl **SAHF** (store AH into flags) realisiert die entsprechende Transportoperation für die Gegenrichtung. Die **IN**- und **OUT**-Befehle sind 8- und 16-Bit-Transportbefehle zwischen **AL** oder **AX** und Ein-/Ausgabekanälen. Dabei kann die Kanaladresse als Bytekonstante im Befehl oder als Wort-Variable im Register **DX** angegeben werden.



Mit den folgenden Befehlen werden einzelne Binärstellen des Flagregisters eingestellt:

CLC (clear **CF**)	;<**CF**>=**0**
CMC (complement **CF**)	;<**CF**>=<$\overline{\textbf{CF}}$>
STC (set **CF**)	;<**CF**>=**1**
CLD (clear **DF**)	;<**DF**>=**0**
STD (set **DF**)	;<**DF**>=**1**
CLI (clear **IF**)	;<**IF**>=**0**, Sperren ;maskierbarer ;Interrupts
STI (set IF)	;<**IF**>=**1**, Freigabe ;maskierbarer ;Interrupts

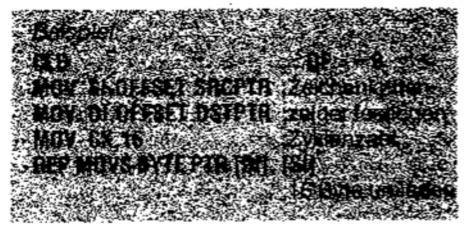
Eine andere Möglichkeit zur Einstellung aller Flags besteht in der Abspeicherung des gewünschten Bitmusters im Stack und dessen Übernahme in das Flagregister mit dem Befehl **POPF**.

5.3.2 Zeichenkettenbefehle

Zeichenkettenbefehle dienen dem Umladen und Vergleich von bis zu 64 KByte großen Speicherbereichen bzw. der Zeichensuche und Zeicheneintragung in Speicherbereichen. Für die entsprechenden String-*Einzeloperationen*

MOVS	move string
CMPS	compare string
STOS	store string
LODS	load string
SCAS	scan string

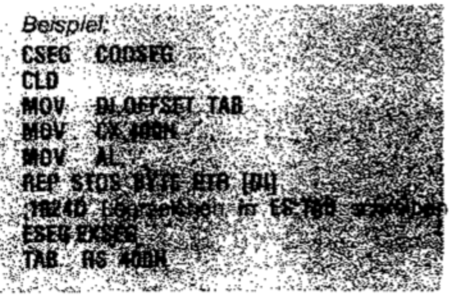
kann durch die Angabe des Wiederholungspräfix **REP** eine zyklische Abarbeitung festgelegt werden, für die das Register **CX** die Zyklenzahl enthält. Mit dem Präfix **REP** wird bei den Befehlen **CMPS** und **SCAS** eine zweite Wiederholbedingung des zyklischen Ablaufes in Abhängigkeit vom Inhalt des Z-Flags durchgeführt. Es erfolgt dabei eine automatische Wiederholung unter der Bedingung **CX** ungleich Null und <**ZF**>=1.
Einheitlich für alle Zeichenkettenbefehle gilt, daß die *Quellzeichenkette* mit der Offsetadresse im Register *SI* und der Segmentadresse in *DS* und die *Zielzeichenkette* mit der Offsetadresse in *DI* und der Segmentadresse in *ES* definiert ist. Dadurch sind Zeichenkettenoperationen mit unterschiedlichen Speichersegmenten möglich.
Für eine eindeutige Übertragung ist die Angabe **BYTE PTR** bzw. **WORD PTR** erforderlich.
Der Befehl **MOVS** überträgt ein Byte oder Wort aus der Quellzeichenkette in die Zielzeichenkette. Bei Abarbeitung des Befehls werden die Zeiger *SI* und *DI* für beide Zeichenketten in Abhängigkiet vom Datenformat um *1* und 2 verändert. Ein Zeigerinkrement oder -dekrement erfolgt in Abhängigkeit vom Wert des **D**-Flags (**DF=1** Dekrement, **DF=0** Inkrement).
Der **MOVS**-Befehl beeinflußt keine Flags.



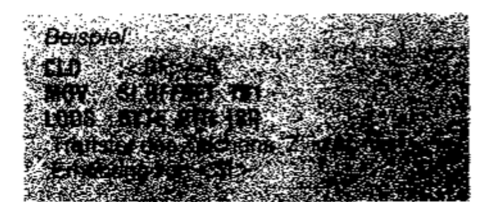
Im Befehl **CMPS** werden die Quell- und Zielzeichenkette byte- oder wortweise verglichen und die Zeiger wie beim **MOVS**-Befehl verändert. Das Ergebnis des Vergleichs steht in den Flags **AF**, **CF**, **OF**, **PF**, **SF** und **ZF**. Der von der Z-Flagbedingung abhängige zyklische Vergleich ergibt die Zeiger **DI**, **SI** auf die nächste Adresse der ersten Zeichenabweichung oder -übereinstimmung.
REP - **REPZ**, Wiederholung bei <**ZF**>=1
REPNZ, Wiederholung bei <**ZF**>=0



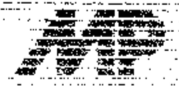
Mit dem Befehl **STOS** kann der Zielbereich im Extrasegment mit in **AL** oder **AX** vorgegebenen Bytes/Wörtern beschrieben werden.



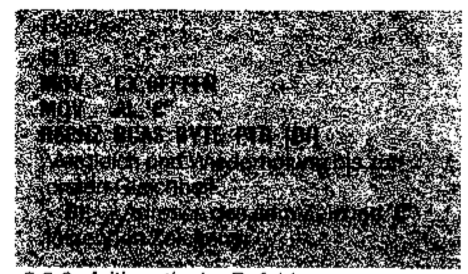
Die umgekehrte Operation führt der Befehl **LODS** aus. Ein Byte/Wort wird aus der Quellzeichenkette vom Datensegment in AL/AX übertragen.



Für die Suche eines in **AL/AX** vorgegebenen 8-/16-Bit-Zeichens in einer Zielzeichenkette im Extrasegment ist der Befehl **SCAS** in zyklischer Ausführung mit dem Präfix REPNZ geeignet.



Kurs



5.3.3 *Arithmetische Befehle*

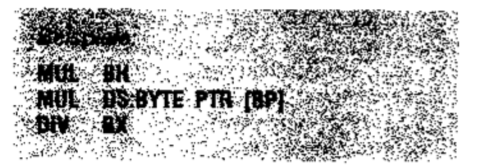
Die arithmetischen Operationen realisieren die vier Grundrechenarten für 8-/16-Bit-Dualdarstellungen und zweistellige Dezimalzahlen in unterschiedlichen Versionen. Operanden können in 8-/16-Bit-Registern oder im Speicher vorgegeben werden. Das Ergebnis kann in jedem dieser Register oder im Speicher abgelegt werden.
Im Ergebnis arithmetischer Operationen werden zusätzlich die folgenden Flageinstellungen realisiert:

CF = '1', wenn Überlauf bei Addition oder Unterlauf bei Subtraktion für das gesamte Format auftritt
ZF = '1', wenn Ergebniswert = 0
SF = '1', wenn höchstwertigstes Bit des Ergebnisses = 1
PF = '1', wenn Parität der unteren 8 Bit des Ergebnisses gerade
AF = '1', wenn bei Addition Überlauf oder bei Subtraktion Unterlauf zwischen den unteren Halbbytes des Ergebnisses auftritt
OF = '1', wenn ein Überlauf aus der zweithöchsten Stelle auf die werthöchste Stelle des Ergebnisses entsteht

Die Addition und Subtraktion von Dualdaten erfolgt in den Befehlen **ADD**, **SUB**, **ADC** (addition with carry) und **SBB** (subtraction with borrow) ohne oder mit Berücksichtigung des C-Flags. Negative Zahlen werden in der Zweierkomplementform dargestellt.
Sonderfälle der Addition und Subtraktion sind die **INC**- und **DEC**-Operationen mit der Erhöhung oder Erniedrigung des Inhaltes um 1 von 8-/16-Bit-Registern oder von Dualzahlen im Speicher.
Eine duale Subtraktion wird beim **CMP**-Befehl ausgeführt, wobei das Ergebnis nur die o. a. Flags beeinflußt.

```
Beispiele:
ADD  [SI],CX
CMP  CS:BYTE PTR [DI]+3,'A'
INC  WORD PTR [BX]
```

Der Befehl **NEG** erzeugt die Zweierkomplementdarstellung des Vorgabeoperanden.
Die dezimale Addition und Subtraktion werden über Korrekturoperationen im Anschluß an die dualen Operationen ausgeführt.
Der Befehl **DAA** (decimal adjust for addition) wandelt das duale Additionsergebnis im Register **AL** in eine gepackte Dezimaldarstellung um. Für die 3stellige dezimale Ergebnisdarstellung wird das C-Flag mitbenutzt.
Im Anschluß an die duale Addition einer zweistelligen ungepackten Dezimalzahl im ASCII-Code im Register AX mit einer einstelligen Dezimalzahl in Dualdarstellung erzeugt der Befehl **AAA** (**ASCII** adjust for addition) eine ASCII-Darstellung des Ergebnisses in **AX**.
Die entsprechenden Korrekturoperationen für die Subtraktion lauten **DAS** (decimal adjust for subtraction) und **AAS** (**ASCII** adjust for subtraction).
Eine bedeutende Verbesserung gegenüber der 8-Bit-Prozessor-Technik bietet beim **8086** die Hardwarerealisierung der Multiplikation und Division.
Der Multiplikationsbefehl **MUL** führt eine vorzeichenlose Multiplikation zwischen dem Register **AL** bzw. **AX** und einem Faktor in einem Register oder im Speicher aus. Im Fall einer 8-Bit-Multiplikation wird der über 8 Binärstellen reichende Anteil des Produktes in **AH** abgespeichert. Bei einer 16-Bit-Multiplikation kann das Ergebnis maximal 32 Binärstellen einnehmen, von denen die oberen 16 im Register **DX** abgelegt werden. Ist die Ergebnislänge größer als das Vorgabeformat, werden die Flags **CF**=**OF**='1' gesetzt. Alle anderen Flags sind nicht signifikant.
Der Befehl **IMUL** (integer multiply) verarbeitet vorzeichenbewertete Dualzahlen.
Für die Multiplikation von ungepackten Dezimalzahlen existiert die Korrekturoperation **AAM** (**ASCII** adjust for multiply) für die Nachbehandlung einer 8-Bit-Dualmultiplikation in **AX**.
Bei dem Befehl **DIV** muß ein ganzzahliger Dividend doppelter Länge durch einen Divisor einfacher Länge dividiert werden. Der Dividend wird bei der 8-Bit-Division im Register **AL** mit der Erweiterung in **AH** vorgegeben. Der Quotient steht nach der Operation in **AL** und der Rest in **AH**.
Bei der 16-Bit-Division wird der 32-Bit-Dividend in **AX** mit der Erweiterung in **DX** angegeben. Im Ergebnis stehen der Quotient in **AX** und der Divisionsrest in **DX**. Auf diese Weise können Divisionsbefehle verkettet werden, um Dualzahlen beliebiger Länge durch 8- oder 16-Bit-Zahlen zu dividieren.
Bei der Division wird im Fall eines Ergebnisüberlaufes (divide by zero) ein nicht maskierbarer Software-Interrupt auf eine in der Interrupttabelle stehenden Absolutadresse ausgelöst.
Der Befehl **IDIV** (integer division) führt die entsprechende Divisions-Operationen an vorzeichenbewerteten Operanden aus.



Nach einer Dualdivision von bis zu zweistelligen ungepackten Dezimalzahlen erzeugt der Befehl **AAD** (**ASCII** adjust for division) die **ASCII**-Darstellung des Quotienten.
Für Formatverlängerungen von vorzeichenbewerteten Dualzahlen in **AL** ergänzt der Befehl **CBW** (convert byte to word) das Erweiterungsregister **AH** bei positiven Werten mit **00H**, andernfalls mit **FFH**.
Der entsprechende Befehl **CWD** (convert word to double word) für die 16-Bit-Version füllt das Erweiterungsregister **DX** mit **0000H** oder **FFFFH**.

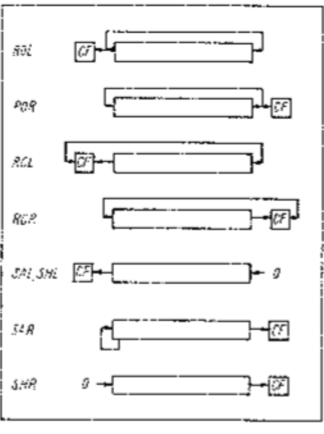


Bild 5.2 Wirkung der Rotations- und Shiftbefehle

5.3.4 *Logikbefehle*

Acht Formen von *Verschiebebefehlen für Rotation und Shift* gestatten Links- und Rechtsverschiebungen von 8- und 16-Bit-Daten in den Registern und im Speicher. Alle Verschiebebefehle können einfach oder mehrfach ausgeführt werden, im zweiten Fall steht im Register **CL** die Zahl der Verschiebungen.
Die *Rotationsbefehle* **ROL** (rotate left) und **ROR** (rotate right) führen zur Links- oder Rechtsrotation des vorgegebenen Operanden. Das auslaufende Bit wird zugleich in das **C**-Flag gesetzt.
Die Befehle **RCL** (rotate through carry flag left) und **RCR** (rotate through carry flag right) schließen das **C**-Flag in die Rotation ein.

```
Beispiele:
RCL AX,1     1 × Linksrotation
             in <AX>
RCR BX,CL    Anzahl der
             Rotationen in <CL>
ROL WORD PTR [SI],1
  1 × Linksrotation des Wortes in
  Adresse DS:<SI>
```

Im Fall der Einzelbitrotationen wird das **O**-Flag als weiteres Flag gesetzt. Für **ROL** und **RCL** ergibt sich **OF** aus einer Antivalenzoperation des werthöchsten Bit des Operanden und des C-Flags, während sich die Antivalenzoperation bei den Befehlen **ROR** und **RCR** auf die zwei werthöchsten Bit des Operanden bezieht.
Die *Shift-Befehle* **SHL** (shift logical left) für die logische Links- und **SHR** (shift logical right) für die logische Rechtsverschiebung unterscheiden sich von den Rotationsbefehlen durch das Nachschieben von '0'-Stellen. Die auslaufende Bitstelle wird nach jedem Zyklus in das **C**-Flag gesetzt.
Arithmetische Shiftbefehle **SAL** (shift arithmetic left) und **SAR** (shift arithmetic right) verarbeiten vorzeichenbewertete Zahlen. Die Verschiebung mit dem Einschieben von '**0**' betrifft nur den Betragsanteil, ohne die werthöchste, das Vorzeichen enthaltende Bitstelle zu verändern.
Die Wirkung der Rotations- und Shift-Befehle faßt Bild 5.2 zusammen.
Die Beeinflussung des **O**-Flags ist auch bei



Kurs

den Shift-Befehlen auf die Einzelbitverschiebung beschränkt. Dabei gelten für die Befehle **SHL** und **SAL** die gleichen Bedingungen wie bei **ROL**, **RCL** und für **SHR** wie bei den Befehlen **ROR**, **RCR**. Abweichend davon wird nach dem Befehl **SAR** für die Einzelbitverschiebung **OF** = '0' gesetzt.
Bei den Shift-Befehlen werden zusätzlich die Flags **PF**, **SF** und **ZF** entsprechend den im Abschnitt 5.3.3 angegebenen Bedingungen gebildet.
Der Befehl **NOT** realisiert das Einerkomplement des Operanden ohne Flagbeeinflussung.



Die Befehle **AND**, **OR** und **XOR** (Antivalenz, exclusive or) führen bitweise logische Verknüpfungen zwischen zwei Operanden aus. Als Operanden sind die für die arithmetischen Befehle angegebenen Kombinationen von Registern, Speicherplätzen und im Befehl angegebenen Konstanten möglich.



Die Flags **PF**, **SF** und **ZF** werden in Übereinstimmung mit den Bildungsregeln und **CF** = **OF** = '0' gesetzt.
Der Befehl **TEST** realisiert eine **AND**-Verknüpfung ohne Ergebnisdarstellung, wobei nur die Flags beeinflußt werden.



5.3.5 Befehle zur Programmablaufsteuerung

Sprungbefehle verändern den Inhalt des Befehlszählers IP und im Falle von Sprüngen zwischen verschiedenen Codesegmenten auch den des Codesegmentregisters.
Bei *unbedingten* Sprüngen werden drei Arten unterschieden:
1. Der Befehl **JMP** für Sprünge *innerhalb* eines Codesegmentes existiert mit direkter und indirekter Adressierung.
Eine *direkt* angegebene Adresse gibt den vorzeichenbehafteten 2-Byte-Abstand gegenüber dem aktuellen Befehlszähler an.



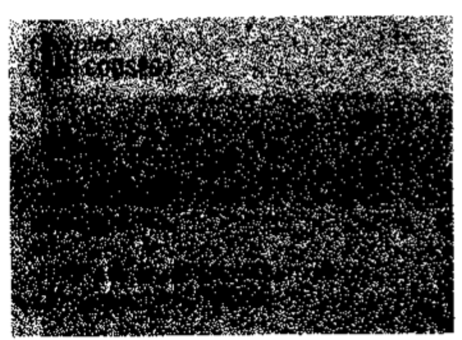
2. Bei der *indirekten* Adressierung kann die Ziel-Offsetadresse in einem der acht 16-Bit-Register oder auf zwei benachbarten Speicherplätzen stehen.



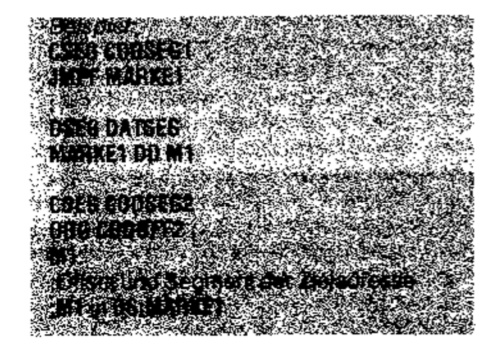
3. Die Kurzform für die direkte Adressierung **JMPS** enthält im Befehl einen vorzeichenbehafteten 1-Byte-Relativabstand.



Intersegmentsprünge zu einem neuen Codesegment mit der Bezeichnung **JMPF** (jump far) müssen zusätzlich die neue Codesegmentadresse enthalten.
Dabei wird für eine *direkte* Adressierung die Zieladresse in der Assemblernotierung einfach in einem anderen Codesegment definiert.



Bei Intersegmentsprüngen mit *indirekter* Adressierung wird die Zieladresse mit je zwei Byte für Offset- und Segmentadresse mit der DD-Anweisung im Speicher abgelegt.



Für *bedingte* Sprünge gibt es nur die Kurzform der direkten Adressierung mit 1-Byte-Relativabstand. Die Ausführung des Sprunges ist von Einzelflags oder Flagkombinationen abhängig. Die Assemblerprogramme lassen für einige Befehle zwei mnemonische Beschreibungen zu:

Name	Sprungbedingung
JZ (zero) = **JE** (equal)	<ZF>=1
JNZ (not zero) = **JNE** (not equal)	<ZF>=0
JB (below) = **JNAE** (not above or equal)	<CF>=1
JNB (not below) = **JAE** (above or equal)	<CF>=0
JS (sign)	<SF>=1
JNS (not sign)	<SF>=0
JP (parity) = **JPE** (parity even)	<PF>=1
JNP (not parity) = **JPO** (party odd)	<PF>=0
JO (overflow)	<OF>=1
JNO (not overflow)	<OF>=0
JBE (below or equal) = **JNA** (not above)	<ZF>=1 oder <CF>=1
JNBE (not below or equal) = **JA** (above)	<ZF>=0 und <CF>=0
JL (less) = **JNGE** (not greater or equal)	<SF> ungleich <OF>
JNL (not less) = **JGE** (graeter or equal)	<SF> gleich <OF>
JLE (less or equal) = **JNG** (not greater)	<SF> ungleich <OF> oder <ZF>=1
JNLE (not less or equal) = **JG** (greater)	<SF> gleich <OF> und <ZF>=0

Eine Sonderform des bedingten Sprunges stellt der Befehl **JCXZ** dar, der nur unter der Bedingung <CX>=0 ausgeführt wird.
Die *Schleifenbefehle* realisieren eine zyklische Ausführung von Sprungbefehlen. Im Befehl **LOOP** dient das Register CX als Schleifenzähler. Der Sprung wird ausgeführt, solange der Inhalt von **CX** größer Null ist. In jedem Zyklus wird der Inhalt von CX um 1 vermindert.



In weiteren Formen von Schleifenbefehlen ist die Ausführung des Sprunges zusätzlich vom Z-Flag abhängig.

Name	Sprungbedingung
LOOPZ = **LOOPE**	<CX>>0 und <ZF>=1
LOOPNZ = **LOOPNE**	<CX>>0 und <ZF>=0

CALL-Befehle zum Aufruf von Unterprogrammen unterscheiden ebenfalls wie bei **JUMP**-Befehlen Sprünge innerhalb eines Codesegments bzw. zwischen verschiedenen Codesegmenten.
Ein Unterprogrammaufruf *innerhalb* eines Segmentes mit dem Befehl **CALL** enthält wie der entsprechende **JMP**-Befehl nur Angaben zum Offset der Zieladresse, da das Codeseg-

ment nicht verändert wird. Bei der Form mit *direkter* Adressierung wird der vorzeichenbehaftete 2-Byte-Abstand angegeben. Bei der *indirekten* Adressierung steht die absolute Offsetadresse in einem 16-Bit-Operandenregister oder auf 2-Byte-Speicherplätzen mit den beim **JUMP**-Befehl erläuterten Adressierungsformen.
Vor der Ausführung des Unterprogrammaufrufs innerhalb eines Segmentes wird die Offsetadresse des dem **CALL**-Befehl folgenden Befehles automatisch im Stack abgespeichert.
Bedingte **CALL**-Befehle sind im System **8086** nicht vorhanden.

Der Befehl **CALLF** (call far) führt zu Unterprogrammaufrufen zwischen den Codesegmenten. Eine direkt oder indirekt angegebene Zieladresse muß wie beim **JMPF**-Befehl den Segment- und Offsetanteil enthalten.
Bei einem **CALL-FAR** wird die aktuelle Rückkehradresse, bestehend aus Codesegment und Instruction-Pointer, automatisch in 4 Byte im Stack abgelegt. Aus den unterschiedlichen Abspeicherungsformen der Rückkehradresse im Stack ergeben sich unterschiedliche *RETURN*-Befehle. Der Befehl **RET** für den Abschluß von mit **CALL** aufgerufenen Unterprogrammen überträgt nur 2 Byte aus dem Stack in den Befehlszähler IP.

Mit **CALLF** angewählte Unterprogramme müssen mit **RETF** abgeschlossen werden, da dieser Befehl neben dem Instruction-Pointer auch das Codesegment aus dem Stack liest. Beide Formen der **RETURN**-Befehle lassen die Angabe von positiven 8- oder 16-Bit-Konstanten zu. Diese geben eine an die **RETURN**-Operation anschließende Erhöhung des SP-Registers an.

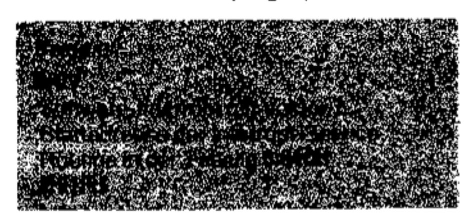
Tafel 5.1 Beispiele für 8086-Befehle

```
                         ;**************************************
                         ;BEISPIELE FUER 8086-BEFEHLE
                         ;**************************************
  1000                   CODSEG1  EQU    1000H    ;CODESEGMENT
  3000                   CODSEG2  EQU    3000H    ;CODESEGMENT
  5000                   ISRSEG   EQU    5000H    ;CODESEGMENT
  2000                   DATSEG   EQU    2000H    ;DATENSEGMENT
  4000                   EXSEG    EQU    4000H    ;EXTRASEGMENT
  0000                   STCKSEG  EQU    0000H    ;STACKSEGMENT
  0000                   INTSEG   EQU    0000H    ;INT.-TAB.-SEGM.
  0600                   STCKP    EQU    0600H    ;STACKPOINTER
  001C                   INTTAB   EQU    001CH    ;INT.-TAB.OFFSET
  0356                   ISROFF   EQU    0356H    ;ISR-OFFSET
  6A39                   CODOFF2  EQU    6A39H    ;COD.-SEGM.2 OFFS.
  15D0                   HALTOFF  EQU    15D0H    ;COD.-SEGM.1 OFFS.
  0021                   COUNT    EQU    0021H    ;ZAEHLER
  57E6                   ADR      EQU    57E6H    ;KONSTANTE
                         ;
  1000                            CSEG   CODSEG1
                                  ORG    0
                         ;TRANSPORTBEFEHLE
0000 8AC6                         MOV    AL,DH
0002 8BD6                         MOV    DX,SI
0004 B8086B                       MOV    AX,6B08H
0007 8BE657                       MOV    CX,ADR
000A 890E0000                     MOV    NAME,CX
000E 8B0E0200                     MOV    CX,MARKE
0012 2E8B0ED215                   MOV    CX,ADR2
0017 BA230F                       MOV    DX,0F23H
001A 8B163DF2                     MOV    DX,.0F23DH
001E 8B77FE              ADR1:    MOV    SI,[BX]-2
0021 887308                       MOV    [BP+DI]+8,BH
0024 368904                       MOV    SS:[SI],AX
0027 BE423E                       MOV    SI,3E42H
002A B564                         MOV    CH,100
002C BB1E00                       MOV    BX,OFFSET ADR1
002F C74403FF37                   MOV    WORD PTR [SI]+3,37FFH
0034 C60101                       MOV    BYTE PTR [BX+DI],1
0037 8D98670A                     LEA    BX,[BX+SI]+0A67H
003B C50B                         LDS    CX,[BP+DI]
003D D7                           XLAT   AL
003E 8735                         XCHG   SI,[DI]
                         ;LADEN DER SEGMENTE
0040 B80000                       MOV    AX,STCKSEG
0043 8ED0                         MOV    SS,AX
0045 B80000                       MOV    AX,INTSEG
0048 8ED8                         MOV    DS,AX
004A B80040                       MOV    AX,EXSEG
004D 8EC0                         MOV    ES,AX
                         ;LADEN STACKPOINTER, INTERRUPT-TABELLE
004F BC0006                       MOV    SP,STCKP
0052 C7061C005603                 MOV    TABOFF,OFFSET ISR
0058 C7061E000050                 MOV    TABSEG,ISRSEG
005E B80020                       MOV    AX,DATSEG
0061 8ED8                         MOV    DS,AX
                         ;E/A-BEFEHLE
0063 E480                         IN     AL,80H
0065 EF                           OUT    DX,AX
                         ;FLAG-BEFEHLE
0066 9F                           LAHF
0067 9E                           SAHF
                         ;MOV-STRING-BEFEHL
0068 FC                           CLD
0069 BE1A00                       MOV    SI,OFFSET SRCPTR
006C BF1200                       MOV    DI,OFFSET DSTPTR
006F B91000                       MOV    CX,16
0072 F3A4                         REP    MOVS BYTE PTR [DI],[SI]
                         ;COMPARE-STRING-BEFEHL
0074 BE0800                       MOV    SI,OFFSET TB1
0077 BF0000                       MOV    DI,OFFSET TB2
007A B96400                       MOV    CX,100
007D F3A6                         REP    CMPS TB2,TB1
                         ;STORE-STRING-BEFEHL
007F BF2200                       MOV    DI,OFFSET TAB
0082 B90004                       MOV    CX,400H
0085 B020                         MOV    AL,' '
0087 F3AA                         REP    STOS BYTE PTR [DI]
                         ;LOAD-STRING-BEFEHL
0089 BE0800                       MOV    SI,OFFSET TB1
008C AC                           LODS   BYTE PTR [SI]
                         ;SCAN-STRING-BEFEHL
008D B043                         MOV    AL,'C'
008F BF0000                       MOV    DI,OFFSET TB2
0092 B9FFFF                       MOV    CX,0FFFFH
0095 F2AE                         REPNZ  SCAS BYTE PTR [DI]
                         ;ARITHMETIK-LOGIK-VERSCHIEDENE-BEFEHLE
0097 010C                         ADD    [SI],CX
0099 2E803D030041                 CMP    CS:BYTE PTR [DI]+3,'A'
009F FF07                         INC    WORD PTR [BX]
00A1 F6E7                         MUL    BH
00A3 3EF66600                     MUL    DS:BYTE PTR [BP]
00A7 F7F3                         DIV    BX
00A9 D1D0                         RCL    AX,1
00AB D3DB                         RCR    BX,CL
00AD D104                         ROL    WORD PTR [SI],1
00AF F610                         NOT    BYTE PTR [BX+SI]
00B1 802752                       AND    BYTE PTR [BX],52H
00B4 234D07                       AND    CX,[DI]+7
00B7 A880                         TEST   AL,80H
00B9 13C3                         ADC    AX,BX
                         ;SOFTWARE-INTERRUPT
00BB CD07                         INT    7
                         ;JUMP-BEFEHLE
00BD E93112         12F1          JMP    $+1234H
00C0 E95DFF         001E          JMP    ADR1
00C3 FFE3                         JMP    BX
00C5 FF6408                       JMP    WORD PTR [SI]+8
00C8 EB09           00D3          JMPS   ADR3
00CA EA396A0030                   JMPF   M1
00CF FF260200                     JMP    MARKE
00D3 FF2E0400            ADR3:    JMPF   MARKE1
00D7 B92100                       MOV    CX,COUNT

00DA 90                  ADR4:    NOP
00DB E2FD           00DA          LOOP   ADR4
                         ;CALL-BEFEHLE
00DD E81000         00F0          CALL   ADR5
00E0 FF160200                     CALL   MARKE
00E4 9A396A0030                   CALLF  M1
00E9 FF1E0400                     CALLF  MARKE1
00ED E80000         00F0          CALL   $+0003
00F0 C3                  ADR5:    RET
                         ;
                                  ORG    HALTOFF
                         ;CPU-STEUERBEFEHLE
15D0 F4                           HLT
15D1 9B                           WAIT
                         ;
15D2 7803                ADR2     DW     0378H
                         ;
                         ;INTERRUPT-SERVICE-ROUTINE FUER INT7
  5000                            CSEG   ISRSEG
                                  ORG    ISROFF
0356 1E                  ISR:     PUSH   DS
0357 9C                           PUSHF
0358 9D                           POPF
0359 1F                           POP    DS
035A CF                           IRET
                         ;
                         ;
  3000                            CSEG   CODSEG2
                                  ORG    CODOFF2
6A39 CB                  M1:      RETF
                         ;
  0000                            DSEG   INTSEG
                                  ORG    INTTAB
001C                     TABOFF   RW     1
001E                     TABSEG   RW     1
                         ;
  2000                            DSEG   DATSEG
                                  ORG    0
0000                     NAME     RW     1
0002 D015                MARKE    DW     HALTOFF
0004 396A0030            MARKE1   DD     M1
0008 5A4549434845        TB1      DB     'ZEICHENKETTE IN DS'
     4E4B45545445
     20494D204453
001A 444153204953        SRCPTR   DB     'DAS IST EIN TEST'
     542045494E20
     54455354
                         ;
  4000                            ESEG   EXSEG
                                  ORG 0
0000 5A4549434845        TB2      DB     'ZEICHENKETTE IN ES'
     4E4B45545445
     20494D204553
0012                     DSTPTR   RS     16
0022                     TAB      RS     400H
                                  END

END OF ASSEMBLY. NUMBER OF ERRORS: 0. USE FACTOR: 12
```

Mit dem speziellen Befehl **INT** (softwareinterrupt) kann direkt eine Interrupt-Service-Routine in einem beliebigen Codesegment aufgerufen werden. Dabei ist im Befehl nur die Angabe des Interrupt-Vektors erforderlich. Die Startadresse der Interrupt-Service-Routine wird dann nach Dekodierung des **INT**-Befehls aus den entsprechenden Plätzen in der Interrupt-Tabelle gelesen.
Vor dem Start der Interrupt-Service-Routine werden das Flagregister und die vollständige Absolutadresse mit Segment und Offset des dem **INT**-Befehl nachfolgenden Befehls automatisch im Stack abgespeichert.
Das *I*-Flag und das *T*-Flag werden gelöscht, so daß weitere Interrupts gesperrt sind.



Der Befehl **INTO** ist ein Spezialfall des INT-Befehls für den interrupt-Vektor 4. Die Ausführung ist von der Bedingung 0-Flag = 1 abhängig.
Die durch Hardware-Interrupt oder Software-Interrupt ausgelösten Routinen müssen durch einen speziellen Rückkehrbefehl **IRET** (interrupt return) abgeschlossen werden.
Der Befehl **IRET** überträgt die im Stack abgelegten Informationen zum Zustand vor dem Interrupt in den Befehlszähler, in das Codesegment und in die Flags. Damit wird die Interruptfähigkeit auch ohne zusätzliche Befehlsnotierung wieder hergestellt.

5.3.6 Befehle für Steuerfunktionen

Mit dem Befehl **WAIT** kann die Programmbearbeitung gestoppt werden. Während der Ausführung dieses Befehls wird der **TEST**-Eingang der **8086-CPU** abgefragt. Dabei verbleibt der Prozessor in einem Wartezustand, solange an diesem Anschluß High-Pegel anliegt.

Der Befehl **HLT** bringt den Prozessor in den Halt-Zustand, der nur durch einen Interrupt wieder aufgehoben werden kann. Durch das 1-Byte-Präfix **LOCK** vor einem Befehl wird der Ausgang '**LOCK**' des Prozessors für die Dauer der Befehlsausführung aktiv. Durch schaltungstechnische Maßnahmen wird in Mehrprozessorsystemen gesichert, daß der **CPU** während eines über mehrere **BUS**-Zyklen reichenden Befehls der **BUS**-Zugriff nicht verlorengeht.
Befehle, die mit dem 5-Bit-Präfix **ESC** (Codierung **11011---**) beginnen, sind für die Zusammenarbeit mit den Koprozessoren des Systems 8086 vorgesehen.
Die **8086-CPU** führt für diese Befehle auch einige Funktionen aus. Die im Befehl angegebene Speicheradresse wird von der **CPU** als physische 20-Bit-Adresse generiert und auf den Adreßbus gelegt. Danach wird durch Steuersignale zwischen den verschiedenen Prozessoren die Kontrolle an einen Koprozessor übergeben, welcher von der selektierten Adresse den Dateninhalt übernimmt. Nach dieser Übernahme werden vom Koprozessor gegebenenfalls benötigte Folgeadressen für weitere Speicherzugriffe eigenständig generiert.
Genauere Darlegungen erfolgen im Abschnitt Koprozessoren.
Beispiele für die **8086**-Befehle bei einem mit **ASM86** assemblierten Quelltext sind in Tafel 5.1 dargestellt.

6. Assemblerprogrammierung mit dem Betriebssystem SCP 1700

6.1 Systemprogramme zur Assemblerprogrammierung

Das Betriebssystem **SCP 1700** enthält eine Reihe von Softwarekomponenten, welche die Assemblerprogrammierung unterstützten. Man kann den Entwicklungsvorgang in 3 Phasen einteilen:

1. Phase: Editieren der Programme
Hilfsmittel:
- Editor **ED**
- Textverarbeitungsprogramme **WS** und **TP**

2. Phase: Assemblieren und Maschinencodeerzeugung
Hilfsmittel:
- Absolutassembler **ASM86**
- Relativassembler **RASM86**
- Filegenerierprogramm **GENCMD**
- Linker **LINK86**

3. Phase: Testung und Fehlersuche
Hilfsmittel:
- Debugger **DDT86**
- symbolischer Debugger **SID86**

Folgende Hilfsmittel sind als Minimalausstattung anzusehen:
- **ED**
- **ASM86**
- **GENCMD**
- **DDT86** oder **SID86**.

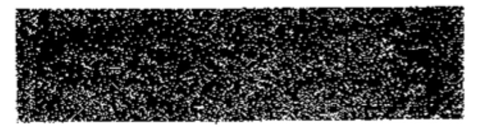
Unter Nutzung der 8-Bit-Technik gibt es direkte Äquivalente:
- **ED**
- **ASM86**
- **GENCMD**,

so daß Vorarbeiten auf der Basis der Crosssoftwareentwicklung, die im System **SCP 1700** oder zu dessen Implementierung genutzt werden können, möglich sind. In Bild 6.1 ist übersichtsmäßig der Entwicklungsweg von der Problemlösung bis zum fertigen Programm dargelegt.

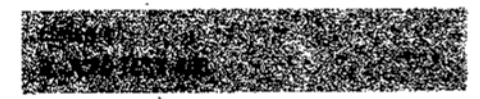
Bild 6.1 Softwareentwicklungszyklus

6.2 Der Zeileneditor ED

Der Zeileneditor **ED** läßt sich im Betriebssystem **SCP 1700** universell einsetzen, so z. B. zum Editieren von Assemblerquelltexten, von Hochsprachprogrammen (z. B. **BASIC**, **PASCAL**, **C**) und Texten allgemeiner Art.
Für die Arbeit mit **ED** empfiehlt es sich vor Arbeitsbeginn, das System **SCP 1700** auf das aktuelle Arbeitslaufwerk einzustellen.



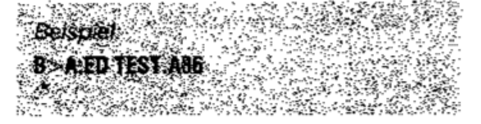
Hierbei können weitgehend Fehler vermieden werden, die durch das ungewollte Beschreiben der Systemdiskette o. ä. entstehen können. Weiterhin erhöht sich die Arbeitsgeschwindigkeit der Dienstprogramme bei der Bearbeitung der Areitsdateien. Zum Starten von **ED** muß das Kommandowort "**ED**" **mit** einer Dateispezifikation **ohne** Sonderzeichen eingegeben werden.



Ein Vergessen der Angabe der Dateispezifikation kann zu unkontrollierten Schreibzugriffen auf der Diskette führen, die unter Umständen das Directory zerstören können! Für besondere Fälle ist es erforderlich, die neue Datei unter einem anderen Namen und eventuell auf einem anderen Diskettenlaufwerk ablegen zu müssen. Die Kommandoeingabe sieht dann wie folgt aus:

Beispiel:
B>A:ED TEST.A86 d:TEST1.A86

Dabei gilt die Voraussetzung, daß die zweite Datei noch nicht existent ist! ED meldet sich nach dem Start mit seinem Kommandoprompt und erwartet eine Kommandoeingabe:



Literatur
/1/ Rextor, R.; Alexy, G.: Das 8086/8088 Buch, Programmieren in Assembler und Systemarchitektur. te-wi Verlag, München 1982
/2/ Jorke, G.; Lampe, B.; Wengel, N.: Arithmetische Algorithmen der Mikrorechentechnik. VEB Verlag Technik, Berlin 1989
/3/ Intel – Applikation Note AP-59
Using the 8259A Programmable
Interrupt Controller.
September 1979

wird fortgesetzt

Unsere Gesprächspartner



Prof. Dr. rer. oec. H. Tzschoppe (1934) studierte an der TH Dresden (1954–1960) Ingenieurökonomie. Bis 1977 in leitender Funktion des Kombinates Robotron tätig. Anschließend Berufung zum ordentlichen Professor an der IH Dresden. Seit 1986 Direktor des Informatik-Zentrums.



Prof. Dr. sc. techn. R. Giesecke (1936) studierte von 1954–1960 an der TH Dresden Flugfunktechnik. Bis 1975 leitender Mitarbeiter im Kombinat Robotron. Anschließend Dozent an der IH Dresden; seit 1982 ordentlicher Professor. Am Informatik-Zentrum stellvertretender Direktor für Weiterbildung.



Prof. Dr. sc. techn. H. Löffler (1934) studierte an der TH Dresden von 1952–1958 Elektro- und Kerntechnik. Von 1969–1971 als Hochschullehrer an der Technischen Staatsuniversität Santiago/Chile. Anschließend Hochschullehrer an der TU Dresden. 1974 Berufung zum ordentlichen Professor. Am Informatik-Zentrum stellvertretender Direktor für Wissenschaftskooperation und Forschung.



Prof. Dr. sc. techn. H. Pieper (1934) studierte an der TU Dresden Hochfrequenztechnik. Seit 1969 an der IH Dresden tätig. 1987 Berufung zum ordentlichen Professor. Leiter des Wissenschaftsbereiches Grundlagen der Informatik am Informatik-Zentrum.

Mittel und Methoden der Informatik aktiver und umfassender nutzen

Im Oktober 1986 wurde durch den Minister für Hoch- und Fachschulwesen der DDR, Prof. Böhme, im Auftrag der Partei- und Staatsführung der DDR die Bildung des Informatikzentrums vollzogen.
Wir wollten gern Näheres über Ziele und Aufgaben des Informatikzentrums des Hochschulwesens wissen und sprachen aus diesem Grund mit der Leitung des Zentrums.

MP: *Welche Gründe führten zur Bildung des Informatikzentrums?*

Prof. Tzschoppe: Es ist objektive Erkenntnis, daß die Informatik in zunehmendem Maße die Rolle einer Basiswissenschaft einnimmt, daß alle informationsverarbeitenden Technologien, wie CAD/CAM, flexible Produktionsautomatisierung, Büroautomatisierung, d. h., man kann sagen fast alle Schlüsseltechnologien, ohne Ergebnisse der Informatik kaum auskommen. Ausgehend von dieser Tatsache erweist es sich als notwendig, im Prozeß der Beschleunigung des wissenschaftlich-technischen Fortschritts, auch unter Berücksichtigung der Auseinandersetzung mit dem kapitalistischen System, gerade auf diesem wichtigen Gebiet zu einer höheren Effektivität in der Ausbildung und der Forschung zu gelangen. Es war deshalb folgerichtig, die hier in Dresden angesiedelten Kapazitäten der ehemaligen Ingenieurhochschule und der Technischen Universität, die auf dem Gebiet der Informationsverarbeitung tätig sind, zu konzentrieren. Ein weiterer Grund bestand darin, daß im Raum Dresden bedeutsame Potentiale der Industrie auf dem Gebiet der Herstellung von Informationsverarbeitungstechnik, aber auch der Mikroelektronik, wirken. Damit kann über eine Erhöhung des Potentials in der disziplinären Forschung durch das Hochschulwesen auch eine konzentrierte Zusammenarbeit mit diesen Hauptpartnern der Industrie organisiert werden; besser als über die vormals zersplitterten Kapazitäten.

MP: *Können Sie etwas über Aufbau und Struktur des Informatikzentrums (IZ) sagen?*

Prof. Tzschoppe: Ausgehend von der Einheit von Forschung und Lehre, wobei das Primat in einer Hochschuleinrichtung die Lehre hat, wurden die Strukturen nach den Fachrichtungen gebildet, die wir in der Grundstudienrichtung haben. Es gibt also die folgenden Wissenschaftsbereiche: Theoretische Informatik, Systemsoftware, Angewandte Informatik und Rechnersysteme.
Außerdem hat das IZ einen WB Grundlagen der Informatik, in dem zwei Aufgaben angesiedelt sind: die Grundlagenausbildung aller Ingenieure und Ökonomen der Technischen Universität auf dem Gebiet der Informatik und die Grundlagenausbildung physikalisch-elektronischer Natur aller Informatikstudenten. Ein Novum ist, daß wir hier am Zentrum eine relativ große Kapazität an Hardwarefachleuten besitzen, weil wir davon ausgehen, daß eine moderne Informatik zur disziplinären Informatikforschung die unmittelbare Einheit von Hardware und Software im Forschungsprozeß braucht. Sie läßt sich nicht über Kooperationslinien und andere Regelungen interdisziplinär organisieren, sondern sie muß integriert sein. Dabei verfügen wir über ein leistungsfähiges eigenes Rechenzentrum. Die Informatikforschung wird also innerhalb des Zentrums mit eigener Rechentechnik realisiert. Über das Rechenzentrum wurden Bedienlabors aufgebaut, die von den entsprechenden Wissenschaftsbereichen genutzt werden. Neu ist in der Struktur dieses Zentrums auch, daß ein eigener wissenschaftlicher Gerätebau für Informationstechnik/Informatik existiert, wo zukünftig experimentelle Muster entstehen sollen, z. B. Controller, Spezialprozessoren, die unmittelbar in der Informatikforschung benötigt werden.
Neu ist auch die Bildung von drei Stellvertreterbereichen, für Wissenschaftskooperation/Forschung, für Weiterbildung und für Erziehung und Ausbildung.
In diesen Stellvertreterbereichen sind zunächst einmal die wesentlichsten Funktionen institutionell angesiedelt, die mit der Zentrumsfunktion zu tun haben, also Aufgaben der Wissenschaftskooperation im Hochschulwesen, der Softwarekoordinierung, der Ausarbeitung von Rechnerstrategien für das Hochschulwesen und der Koordinierung der Weiterbildung im Hochschulwesen auf dem Gebiet der Informatik.

MP: *Wie charakterisieren Sie die Aufgaben des IZ innerhalb der TU und des Hochschulwesens?*

Prof. Tzschoppe: Kommen wir zunächst zur Einordnung des IZ an der Universität. Drei große Gebiete bestimmen die Aufgaben des IZ: Zum ersten einen bedeutsamen Beitrag für die gesamte Informatikausbildung der Ingenieure und Ökonomen an der TU Dresden zu leisten, neben der Ausbildung der Spezialisten in der Grundstudienrichtung Informatik, für die das IZ verantwortlich ist. Das heißt, das Zentrum hat an der Universität die Aufgabe und Verantwortung, die Grundlagenausbildung aller Ingenieure und Ökonomen abzusichern, inhaltlich zu gestalten und für die einzelnen Fakultäten und Sektionen die vertiefende Ausbildung – also die spezialisierte Informatikausbildung im Sinne entwicklungsorientierter Informatiker – in den Fachrichtungen zu übernehmen. Das Zentrum hat also eigene Beiträge zum Beispiel für die Verfahrenstechnik und das Maschinenbauingenieurwesen in der Grundlagenausbildung zu bringen. Dabei ist natürlich eine Kooperation erforder-

lich, um z. B. bei 10000 Studenten 150 Stunden Grundlagenausbildung realisieren zu können. Aus diesem Grund kooperieren wir mit der Sektion Mathematik und dem Rechenzentrum der TU. Die Verantwortung für die inhaltliche Gestaltung der Lehrkomplexe liegt natürlich beim Informatikzentrum. Die vertiefende Ausbildung dient dann der Realisierung spezialisierter Lehrveranstaltungen, beispielsweise auf dem Gebiet der Softwaretechnologie, der Datenbanken, der Bildverarbeitung usw. entsprechend dem Profil der Fachrichtungen.

Zweitens hat das IZ an der TU Dresden bedeutsame Beiträge zur Gestaltung von Ingenieurarbeitsplätzen und zur Entwicklung der flexiblen Produktionsautomatisierung zu leisten – im Sinne von Grundlagenarbeiten zur automatisierten Fabrik der Zukunft (Rechnernetzgestaltung). Damit ordnet sich das IZ in Hauptforschungslinien der Universität ein, wobei die Gesamtverantwortung bei anderen Sektionen liegt, das IZ aber bestimmte interdisziplinäre Beiträge zu bringen hat.

Eine dritte Aufgabe betrifft die Weiterbildung innerhalb der Universität. Das IZ hat Beiträge für die Weiterbildung des wissenschaftlichen Nachwuchses von der Universität zu leisten, um über die Grund- und vertiefende Ausbildung Nachwuchskadern hinausgehende Kenntnisse zu vermitteln, die dann zu Graduierungen führen und bis zur Vorbereitung auf bestimmte Berufungsgebiete, z. B. im Maschineningenieurwesen, reichen.

Prof. Giesecke: Gestatten Sie mir noch eine Bemerkung zur inhaltlichen Gestaltung der Informatikausbildung. Im Vorfeld der Gründung des IZ wurde vom Beirat Informatik inhaltlich ein neuer Studienplan erarbeitet, der die Ausbildung in vier Fachrichtungen einer neu profilierten Grundstudienrichtung Informatik vorsieht. Diese Fachrichtungen entsprechen weitestgehend den volkswirtschaftlichen Bedürfnissen, das heißt, die inhaltliche Gestaltung dieser Fachrichtungen ist mit den wichtigsten Kombinaten und Betrieben diskutiert und abgestimmt worden. Die Bedeutung des IZ im Hochschulwesen wird auch deutlich, wenn wir den Ausbildungsumfang betrachten. Wir bilden im IZ in allen vier Fachrichtungen aus und sind verantwortlich für die Immatrikulation von jährlich zwei Drittel aller Studenten, die in der DDR in der Grundstudienrichtung Informatik studieren. Weitere Bildungseinrichtungen sind die Wilhelm-Pieck-Universität Rostock, die TU Magdeburg und die TU Karl-Marx-Stadt.

MP: *Nimmt das IZ Einfluß auf die Informatikausbildung innerhalb des Hochschulwesens, z. B. eines zukünftigen Ökonomen, der an der Hochschule für Ökonomie in Karlshorst studiert?*

Prof. Tzschoppe: Es gibt beim Minister für Hoch- und Fachschulwesen einen wissenschaftlichen Beirat für Informatik, der den Minister und seine Organe berät und zur Bildungs-, Forschungs- und Weiterbildungsstrategie der Informatik im Hochschulwesen Empfehlungen ausarbeitet. Der Beirat hat die Bildungskonzeption Informatik mit erarbeitet. Vom Beirat wurden drei Kategorien empfohlen: Alle Ingenieure und Ökonomen sollen eine Grundlagenausbildung erhalten.

In den technischen und ökonomischen Disziplinen sollen etwa 15 bis 20 Prozent aller Studenten einer Fachrichtung eine vertiefende Informatikausbildung erhalten im Sinne entwicklungsorientierter Informatiker in den Fachrichtungen. Also im Maschineningenieurwesen sollen es die Kader sein, die die CAD/CAM-Anwendersoftware in ihrem zukünftigen Maschinenbaubetrieb selbst entwickeln. Und die dritte Kategorie sind die Spezialisten, eben die Studenten der Grundstudienrichtung Informatik.

Das ist so vom Minister aufgegriffen und zunächst einmal für alle technischen Disziplinen und Studienrichtungen als Orientierung herausgegeben worden.

Die wissenschaftlichen Beiräte der technischen Wissenschaften haben sich zu dieser Grundorientierung bekannt. Im gleichen Maße hat sich der Beirat Wirtschaftswissenschaften dieser Konzeption angeschlossen, hat sich etwas modifiziert, und die Ausbildung ebenfalls nach diesen Kategorien organisiert. Die Realisierung ist Aufgabe der jeweiligen Beiräte. Der Beirat Informatik unterstützt den Beirat Wirtschaftswissenschaften, was die Grundlagenausbildung betrifft. Für die Informatikausbildung in den Fachrichtungen selbst tragen die jeweiligen wissenschaftlichen Beiräte die Verantwortung.

MP: *Wir haben bereits über die neuen Inhalte der Informatikausbildung gesprochen; können Sie das noch etwas konkreter ausführen, auch was das Verhältnis Hardware/Software betrifft?*

Prof. Pieper: Die frühere Informatikausbildung war ja nahezu vollständig umschrieben mit dem Begriff Software. Es gab den Softwareingenieur schlechthin, und der Hardwareingenieur war der Informationstechniker. Mit der Einführung des neuen Informatikstudienplanes hat man der inzwischen vollzogenen Entwicklung Rechnung getragen, nämlich der zunehmenden Integration beider Gebiete. Dabei wurde die Hardwareausbildung neu durchdacht. Wir haben jetzt eine bemerkenswerten Bereich überlappenden Wissens und Könnens. Das heißt, so wie wir heute verlangen, daß der Informationstechniker in der Lage sein soll, Programme zu schreiben, so wollen wir in einer gewissen Analogie auch erreichen, daß der Informatiker imstande ist, ein für seinen Problemkreis optimales Rechnersystem zu konfigurieren und zu realisieren.

Prof. Giesecke: Wir tragen dieser von Prof. Pieper deutlich gemachten Entwicklung insbesondere durch die Bildung eines WB Grundlagen der Informatik Rechnung. Gleichzeitig haben wir aber auch dafür Sorge getragen, daß diese enge Verflechtung von Hard- und Software sich auch in den anderen WB widerspiegelt, das heißt, dort sind sowohl Kollektive, die auf dem Gebiet der Hardware zu Hause sind als auch solche, die die Softwareerarbeitung beherrschen. Dadurch ergeben sich Möglichkeiten für eine neue Qualität der Ausbildung auf dem Gebiet der Informatik.

Prof. Löffler: Ich möchte das ganz konkret sagen. Früher gab es beim Grundstudium Informationsverarbeitung z. B. keine Lehrveranstaltung physikalische elektronische Grundlagen der Informatik. Jetzt gibt es dieses Fach und drei Semester Lehrveranstaltung Rechnersysteme für alle Studenten, egal welcher Fachrichtung.

Außerdem muß ich generell etwas gegen die Hard- und Softwaretrennung sagen. Die Ausbildung ist auf die Rechnerarchitektur bezogen, und da läßt sich überhaupt nicht zwischen Hardware und Software unterscheiden. Zum Beispiel erhalten die Studenten in dieser Lehrveranstaltung die Assemblerprogrammierung als integralen Bestandteil eines Rechnersystems hardwarenah vermittelt. Alle Studenten erhalten bereits in den ersten Semestern Softwaretechnologie/Programmierungstechnik, was früher viel später gelehrt wurde. Damit kann auf einer einheitlichen Basis nach etwa zwei Jahren die eigentliche Spezialisierung beginnen.

MP: *Welche Aufgaben hat das IZ im Rahmen der Weiterbildung?*
Wie ordnen sie sich ein in die Möglichkeiten der Weiterbildung, die das Schulungszentrum von Robotron, die Weiterbildungsakademie des DVZ und die KDT bieten?

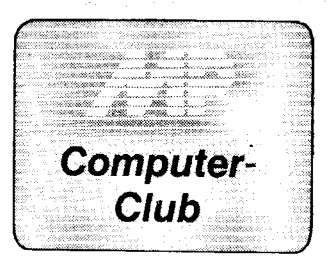
Prof. Giesecke: Die Informatik und/oder Computertechnik gehören zu den Wissenschaftszweigen, die einer sehr kurzen Innovationszeit unterliegen. Wenn man betrachtet, was sich in den letzten 10 Jahren auf unserem Fachgebiet getan hat, so muß man deutlich die Differenz zur Kenntnis nehmen zwischen der Ausbildung von vor 10 Jahren bzw. 15 Jahren und dem Wissen, was heute international und national vorhanden ist. Das heißt, schon aus dieser Situation heraus ist die Notwendigkeit zur Weiterbildung gegeben. Wir verstehen diese Art von Weiterbildung so, daß die Informatiker zukünftig nach bestimmten Zeiten wieder an die Universitäten zurückkehren, um mit dem neuesten Erkenntnisstand vertraut gemacht zu werden. Das ist eine wichtige Aufgabe für uns. Ich muß allerdings deutlich sagen, daß uns auf dem Gebiet der Weiterbildung noch eine ganze Reihe anderer Aufgaben drücken, die gelöst werden müssen. In den letzten Jahren hat die Bereitstellung von Rechentechnik einen enormen Aufschwung genommen. Dieser Aufschwung hat eine gewisse Diskrepanz zwischen den Möglichkeiten der Betriebe, die moderne Technik aufzunehmen und sie auch sofort effektiv einzusetzen, sichtbar gemacht. Es gibt ja bekanntermaßen genaue Forderungen, die den Einsatz der Rechentechnik und ihre effektive Auslastung betreffen. Die von mir genannte Diskrepanz kann u. a. nur durch eine schnelle tiefgründige Qualifizierung der erforderlichen Kader abgebaut werden. Es sind nicht nur Informatiker zu qualifizieren, sondern Kader, die seit Jahren in der Industrie auf ihrem Fachgebiet tätig sind und Spezialkenntnisse der Informatik benötigen. Die Betriebsakademien, das Schulungszentrum von Robotron und andere Bildungseinrichtungen wie die KDT sind allein nicht ausreichend. Hier muß auch das Hochschulwesen und im besonderen das IZ Unterstützung geben.

Prof. Löffler: Solche Aufgaben sind aber nicht hochschultypisch. Unsere Hauptaufgabe ist, die neuen Technologien zu verbreiten und die alten Informationsverarbeiter und Mathematiker, die auf dem Gebiet arbeiten, mit solchen innovativen Techniken wie Netztechnologie, künstliche Intelligenz, Softwaretechnologien auszubilden. Das sind Inhalte, die wir vor 10 Jahren noch nicht hatten. Die betriebsakademietypische Ausbildung ist nicht unsere Hauptaufgabe. Hier geben wir ökonomische Unterstützung auf Grund von aktuellen Notwendigkeiten. An der Hochschule muß man 10 Jahre vorausdenken. Wir haben jetzt eine solche Ausbildung. Die Absolventen, die vor 10 Jahren die Hochschule verlassen haben, müssen fortgebildet werden.

Prof. Tzschoppe: Wir brauchen einen ausgewogenen Beitrag zu beidem. Bis 1990 sollen 500000 Arbeitskräfte auf dem Gebiet CAD/CAM ausgebildet werden, das können die Kombinate Robotron, DVZ und die KDT nicht allein bewältigen, es muß aber getan werden. Zur Zeit ist dafür noch nicht die zweckmäßigste Form der Weiterbildung gefunden. Ich glaube, hier muß gemeinsam mit Robotron nach dem Prinzip des Schneeballs ausgebildet werden. Wir bilden nur einen bestimmten Kern aus, und dieser Kern wird in seinem Betrieb weitere Personen schulen. Anders werden wir diese umfangreiche Aufgabe nicht schaffen.

Prof. Giesecke: Lassen Sie mich abschließend noch einige Bemerkungen zum Fernstudium als eine weitere Säule der Weiterbildung machen. Es wurde auch inhaltlich verändert. Am IZ haben wir derzeit jährlich 350 bis 400 Prozent Bewerbungen für ein Fernstudium auf dem Gebiet der Informatik. Da das IZ zur Zeit die einzige Einrichtung im Hochschulwesen ist, die Fernstudenten auf diesem Gebiet ausbildet, wird das Problem deutlich, vor dem wir stehen. Analysen zeigen aber, daß nur ein kleiner Teil der Bewerber Kader sind, die Spezialisten der Informatik werden müssen. Es handelt sich meist um Mitarbeiter der Industrie, die bereits über Teilkenntnisse verfügen, und die jetzt auf Grund der im Betrieb vorhandenen Technik tiefer in die Informatik eindringen müssen. Wir brauchen also eine informatikrelevantere Ausbildung in den Fachrichtungen des Fernstudiums der anderen Grundstudienrichtungen. Und das muß auch entsprechend publiziert werden.

Außerdem bin ich der Meinung, daß sich die bereits genannten Universitäten, die auf dem Gebiet der Informatik ausbilden, zukünftig dem Fernstudium für Informatiker anschließen müssen.



Reassembler für EMR U 881/882 unter U-880-Systemen

Bei der Erprobung von EMR-Software, die in der Regel mit Wirtsrechnern des U-880-Systems (BC, PC, K1520 usw.) und EPROM-RAM-Adaptern erfolgt, sind oft Änderungen in den Maschinencodeprogrammen erforderlich, für die nicht gleich Editor und Assembler eingesetzt werden. Hier erweist sich der Einsatz eines einfachen Monitors mit einem entsprechenden Reassembler für den jeweiligen Prozessortyp als günstig.

Programmbeschreibung

Das Reassemblerprogramm (**Bild 1**) bezieht sich in seiner Mnemonik auf /1/. Es übersetzt einen Befehl ab der Adresse, die in 4558 steht. Alle genutzten Register werden gerettet. Nach Abarbeitung des Programms steht in 4558 die Adresse des nächsten Befehls. Die Register 240–255 werden mit ihren Kurzbezeichnungen übersetzt. Für nichtinterpretierbare Befehle wird ??? ausgegeben. Register, die im Bereich von 128 bis 223 (verbotener Bereich) liegen, werden ebenfalls als ??? ausgegeben. Für die Register 224–239 wird in der Mnemonik für den entsprechenden „langen" Befehl ein „kurzer" Arbeitsregisterbefehl ausgegeben, z. B. existiert ein Befehl **DEC R1** nicht, sondern nur der Befehl **DEC 001**. Der entsprechende Code wäre **00 01**. Würde als Code **00 E1** angegeben, läge das angegebene Register im verbotenen Bereich, der EMR blendet das obere Nibble der Registeradresse aus und übernimmt dies aus dem Registerpointer, macht also quasi aus einem Registerbefehl einen Arbeitsregisterbefehl. Da die üblichen EMR-Assembler diese „Quasiarbeitsregisterprogrammierung" durch die Eintragung von **E** im oberen Nibble der Registeradresse unterstützen, werden auch diese Befehle durch den Reassembler entsprechend übersetzt. Relativsprünge werden mit Angabe der absoluten Zieladresse übersetzt.

Unterprogramme

Der Reassembler benötigt für seine Arbeit einige universelle Unterprogramme. Bis auf das Unterprogramm zur Bildschirmübergabe sind alle Unterprogramme implementiert. Da die Bildschirmverwaltung sehr unterschiedlich sein kann, beziehen sich sämtliche UP-Aufrufe auf die Adresse 455A, die als Absprung in das jeweilige Bildschirmprogramm dient. Das entsprechende Bildschirmunterprogramm muß den ASCII-Kode verarbeiten, insbesondere die Zeichen **0D** = *Kursor zurück an den Anfang der Zeile* und **0A** = *Kursor eine Zeile tiefer (einschließlich Bildschirmrollen)*. Die sonstigen Unterprogramme sind Tafel 1 zu entnehmen.

Implementierungshinweise

Das Programm wurde für einen OEM-Rechner mit eigenem Monitor als RAM-Version geschrieben, dort kann im Reassemblerbetrieb zwischen U 881/882- und U 880-Reassembler umgeschaltet werden. Der Reassembler wurde außerdem auf BC A 5120/30 und PC 1715 unter CP/M-Steuerung eingesetzt. Soll das Programm als EPROM-Version genutzt werden, muß die Zwischenspeicheradresse für den Adreßmerker geändert werden. Dies muß an den in Bild 1 unterstrichenen Stellen erfolgen (4005 und 431A), ansonsten wird 4558 als Merker verwendet. Das Programm wird auf Adresse 4000 gestartet. Es ist 1376 Bytes lang. Um Eintippfehler zu vermeiden, steht hinter jeder Zeile ein *XX*, der sich aus dem XOR aller Bytes der Zeile bildet. Am Ende erfolgte noch eine CRC-Berechnung nach dem SDLC-Polynom, um eine Kontrolle über das gesamte Programm zu haben.

Carsten Fischer

***Bild 1* Reassemblerprogramm**

```
 CRC (SDLC) = EE40
4000 C5 E5 DD E5 2A 58 45 7E FE E2 CA 21 43 FE E3 CA  *32*
4010 24 42 FE E4 38 23 FE E8 DA 94 42 FE F2 CA 21 43  *D9*
4020 FE F3 CA 24 42 FE F4 CA 21 43 FE F5 CA 94 42 FE  *EA*
4030 F6 CA 21 43 FE F7 CA 21 43 E6 0F FE 08 D2 99 41  *EA*
4040 FE 02 D2 8D 40 06 02 CD 03 44 7E FE 31 20 21 CD  *10*
4050 29 45 53 52 50 20 20 00 23 7E FE 80 38 07 FE F0  *2F*
4060 30 03 C3 26 43 E6 0F CA 6E 42 CD 2C 43 C3 6E 42  *D7*
4070 E6 F0 DD 21 89 44 CD 56 43 CD F6 43 7E FE 30 28  *1F*
4080 05 CB 47 CA 4D 42 23 CD 83 43 C3 18 43 7E CB 7F  *30*
4090 C2 DC 40 E6 0F 06 02 FE 04 38 02 06 03 CD 03 44  *FC*
40A0 7E E6 F0 DD 21 C9 44 CD 56 43 CD F6 43 7E 23 E6  *02*
40B0 0F FE 02 28 17 FE 03 CA 7B 42 FE 04 CA D1 42 FE  *9F*
40C0 05 CA E0 42 FE 06 CA EF 42 C3 F5 42 0E F0 CD DF  *6A*
40D0 43 CD F0 43 0E 0F CD DF 43 C3 18 43 CB 6F C2 6D  *FE*
40E0 41 CB 57 20 58 06 02 CD 03 44 7E CB 77 20 09 CD  *0D*
40F0 29 45 4C 44 45 00 18 07 CD 29 45 4C 44 43 00 7E  *AA*
4100 CB 47 20 05 CD F6 43 18 05 3E 49 CD 5A 45 CD F6  *52*
4110 43 7E 23 E6 11 28 11 FE 01 28 14 F5 CD FC 43 F1  *65*
4120 FE 10 CA CC 40 C3 7B 42 0E F0 CD DF 43 18 05 0E  *EE*
4130 F0 CD DA 43 CD F0 43 CD FC 43 C3 D4 40 FE C7 CA  *0C*
4140 94 42 FE D4 28 0C FE D6 28 0C FE D7 CA 94 42 C3  *22*
4150 21 43 06 02 18 02 06 03 CD 03 44 CD 29 45 43 41  *50*
4160 4C 4C 20 00 7E FE D6 CA E7 41 C3 86 40 E6 0F 06  *F0*
4170 02 FE 04 38 02 06 03 CD 03 44 7E E6 F0 FE B0 28  *43*
4180 0C CD 29 45 43 50 20 20 20 00 C3 AD 40 CD 29 45  *11*
4190 58 4F 52 20 20 00 C3 AD 40 CA 24 42 FE 09 CA 24  *DE*
41A0 42 FE 0A CA 02 43 FE 0B 28 0E FE 0C 28 76 FE 0D  *B1*
41B0 28 21 FE 0E 28 3D 18 51 06 02 CD 03 44 CD 29 45  *8A*
41C0 4A 52 20 20 20 00 CD 4B 43 CD F0 43 23 CD 70 43  *5E*
41D0 C3 18 43 06 03 CD 03 44 CD 29 45 4A 50 20 20 20  *8C*
41E0 00 CD 4B 43 CD F0 43 06 02 23 7E CD 36 45 10 F9  *B5*
41F0 C3 18 43 06 01 CD 03 44 CD 29 45 49 4E 43 20 20  *F0*
4200 00 0E F0 CD DF 43 C3 18 43 06 01 CD 03 44 7E E6  *22*
4210 F0 CB 7F 20 03 C3 26 43 CB BF DD 21 69 44 CD 56  *FF*
4220 43 C3 18 43 06 02 CD 03 44 CD 29 45 4C 44 20 20  *FC*
4230 20 00 7E FE E3 28 43 FE F3 28 50 E6 0F FE 09 28  *6B*
4240 13 FE 0C 28 20 0E F0 CD DF 43 CD F0 43 23 CD 88  *5E*
4250 43 C3 18 43 23 CD 88 43 2B CD F0 43 0E F0 CD DF  *47*
4260 43 23 C3 18 43 0E F0 CD DF 43 CD F0 43 23 3E 23  *17*
4270 CD 5A 45 7E CD 36 45 C3 18 43 23 0E F0 CD DF 43  *06*
4280 CD F0 43 0E 0F CD DA 43 C3 18 43 23 0E F0 CD DA  *79*
4290 43 C3 D1 40 06 03 CD 03 44 CD 29 45 4C 44 20 20  *37*
42A0 20 00 7E 23 FE D7 28 23 FE E4 28 25 FE E5 28 30  *4B*
42B0 FE E6 28 3B FE E7 28 3D FE F5 28 3F 0E F0 CD DF  *F7*
42C0 43 CD F0 43 CD 36 43 23 C3 18 43 CD 36 43 C3 59  *1C*
42D0 42 23 CD 88 43 CD F0 43 2B CD 88 43 23 C3 18 43  *8F*
42E0 23 CD 88 43 CD F0 43 2B CD 83 43 23 C3 18 43 CD  *0B*
42F0 88 43 C3 6A 42 CD 83 43 C3 6A 42 23 CD 83 43 C3  *2B*
4300 D5 42 06 02 CD 03 44 CD 29 45 44 4A 4E 5A 20 00  *82*
4310 0E F0 CD DF 43 C3 C9 41 23 22 58 45 DD E1 E1 C1  *E4*
4320 C9 06 01 CD 03 44 CD 2C 43 C3 18 43 06 03 3E 3F  *7A*
4330 CD 5A 45 10 F9 C9 23 CD 88 43 3E 28 CD 5A 45 2B  *38*
4340 0E 0F CD DF 43 3E 29 CD 5A 45 C9 7E E6 F0 DD 21  *C8*
4350 29 44 CD 56 43 C9 0F 0F 0F 0F 4F 81 81 81 4F 06  *FB*
4360 00 DD 09 06 04 DD 7E 00 CD 5A 45 DD 23 10 F6 C9  *76*
4370 E5 7E 06 00 CB 7F 4F 28 01 05 2B 09 23 23 CD 4F  *EA*
4380 45 E1 C9 3E 40 CD 5A 45 C5 7E FE F0 38 0B DD 21  *BB*
4390 E9 44 E6 0F CD 5A 43 C1 C9 FE E0 38 07 0E 0F CD  *75*
43A0 DF 43 C1 C9 FE 80 38 05 CD 2C 43 C1 C9 D6 64 0E  *C1*
43B0 31 30 03 7E 0E 30 F5 79 CD 5A 45 F1 C1 C5 0E 2F  *C8*
43C0 0C D6 0A 30 FB C6 0A 47 79 CD 5A 45 78 0E 2F 0C  *6E*
43D0 D6 01 30 FB 79 CD 5A 45 C1 C9 3E 40 CD 5A 45 3E  *2D*
43E0 52 CD 5A 45 7E A1 CB 41 20 D3 0F 0F 0F 0F 18 CD  *F3*
43F0 3E 2C CD 5A 45 C9 3E 20 CD 5A 45 C9 CD 29 45 40  *ED*
4400 52 00 C9 E5 3E 04 90 87 4F CD 29 45 0D 0A 00 CD  *77*
4410 4F 45 CD 29 45 20 20 00 7E CD 36 45 23 10 F9 41  *E0*
4420 3E 20 CD 5A 45 10 F9 E1 C9 41 46 20 20 4C 54 20  *32*
4430 20 4C 45 20 20 55 4C 45 20 4F 56 20 20 4D 49 20  *68*
4440 20 5A 20 20 20 43 20 20 20 41 54 20 20 47 45 20  *0E*
4450 20 47 54 20 20 55 47 54 20 4E 4F 56 20 50 4C 20  *1E*
4460 20 4E 5A 20 20 4E 43 20 20 44 49 20 20 45 49 20  *18*
4470 20 52 45 54 20 49 52 45 54 52 43 46 20 53 43 46  *68*
4480 20 43 43 46 20 4E 4F 50 20 44 45 43 20 52 4C 43  *08*
4490 20 49 4E 43 20 4A 50 20 20 44 41 20 20 50 4F 50  *14*
44A0 20 43 4F 4D 20 50 55 53 48 44 45 43 57 52 4C 20  *74*
44B0 20 49 4E 43 57 43 4C 52 20 52 52 43 20 53 52 41  *6D*
44C0 20 52 52 20 20 53 57 41 50 41 44 44 20 41 44 43  *12*
44D0 20 53 55 42 20 53 42 43 20 4F 52 20 20 41 4E 44  *60*
44E0 20 54 43 4D 20 54 4D 20 20 53 49 4F 20 54 4D 52  *7D*
44F0 20 54 31 20 20 50 52 45 31 54 30 20 20 50 52 45  *10*
4500 30 50 32 4D 20 50 33 4D 20 50 30 31 4D 49 50 52  *66*
4510 20 49 52 51 20 49 4D 52 20 46 4C 41 47 52 50 20  *12*
4520 20 53 50 48 20 53 50 4C 20 E3 7E 23 B7 28 05 CD  *CD*
4530 5A 45 18 F6 E3 C9 F5 1F 1F 1F 1F CD 3F 45 F1 F5  *9D*
4540 E6 0F C6 30 FE 3A 38 02 C6 07 CD 5A 45 F1 C9 7C  *B6*
4550 CD 36 45 7D CD 36 45 C9 00 00 C3 00 00 FF FF FF  *88*
```

Tafel 1 **Unterprogrammverzeichnis**

Adresse	Bedeutung
4529	gibt alle dem UP-Aufruf folgenden Byte auf den Bildschirm aus, bis 00 erkannt wird
4536	gibt den Inhalt von A auf Bildschirm aus
454F	gibt den Inhalt von HL auf Bildschirm aus
455A	hier muß der Sprungbefehl in das eigentliche Bildschirmprogramm eingetragen werden, das AF, BC, HL und IX nicht verändern darf

Literatur

/1/ Technische Beschreibung Einchip-Mikrorechner-Schaltkreise U 881/U 882. VEB Mikroelektronik „Karl Marx" Erfurt

KONTAKT

VEB Schiffswerft Rechlin, Boeker Straße, Rechlin (Müritz), 2085; Tel. 24 17

SRAM-4-KByte-Erweiterungsmodul 2–4002 Erweiterungsbaugruppe für den KC 85/1 bzw. KC 87

Dr. Peter Fröhlich, Rudi Sannert
VEB Robotron-Meßelektronik „Otto Schön" Dresden

Der SRAM-4-KByte-Erweiterungsmodul 2–4002 kann für die Speicherung variabler Daten während des Programmablaufes im KC 85/1 bzw. KC 87 eingesetzt werden. Eine vorhandene Spannungskontrollschaltung garantiert, daß diese Daten bei Abschalten des Rechners bzw. Netzausfall erhalten bleiben. Als Speicherschaltkreis wird der CMOS-RAM 85 631 mit einem 4 KByte-Speicher verwendet (8 × US 224 × 20).

Der Adreßdecoder ist so konzipiert, daß mittels Wickelbrücken der gesamte Speicherbereich (0000H–FFFFH) des KC 85/1 bzw. KC 87 in Blöcken zu 2 KByte adressiert werden kann. Dadurch ist es möglich, mehrere Erweiterungsmodule gleichzeitig zu verwenden.

Der Modul wird im Modulschacht des KC betrieben. Beim Einschalten des Kleincomputers erfolgt automatisch eine Spannungskontrolle der NK-Knopfzellen. Unterspannung und der Betriebszustand des Moduls werden optisch angezeigt.

Einige technische Daten sind in Tafel 1 zusammengestellt.

Tafel 1 **Technische Daten**

Speicherkapazität
4 KByte
Speicherschaltkreis
CMOS-SRAM 85 631 (VEB KWH)
Zugriffszeit
≦ 350 ns (typ.)
Betriebsarten
„Lesen" oder „Schreiben"
Datenerhalt
Bei Abschaltung der Betriebsspannung 5P wird der SRAM über eine interne Stützspannungsquelle, die den Datenerhalt sichert, gepuffert.
Stützspannungsquelle
Reihenschaltung von 3 NK-Knopfzellen mit je 1,2 V; 0,225 Ah
Stützspannungsüberwachung
Eine Kontrollschaltung bewertet den Spannungszustand der Batterien für etwa 2 s unmittelbar nach dem Zuschalten der Betriebsspannung 5P und zeigt Unterspannung optisch an.
Stromaufnahme
5P = 5 V ± 5 %, typ. 0,2 A
Schlafstromverbrauch
typ. 30 μA
Abmessungen
180 mm × 100 mm × 20 mm
Masse
etwa 0,35 kg

Aufbau des SRAM-Moduls

Die Schaltung des SRAM-Moduls besteht im wesentlichen aus folgenden Baugruppen (Bild 1)
- CMOS-RAM
- Adreßdecoder
- Stützspannungsüberwachung und -zuschaltung

Der CMOS-RAM befindet sich in einem 24poligen Standard-Hermetikgehäuse. Die gesamte Speicherkapazität beträgt 32 KBit, die in Speicherblöcken von 4 × 1 KByte organisiert ist. Durch Absenken der Betriebsspannung auf Werte bis 2 V wird ein Schlafzustand erreicht, in dem die eingespeicherten Daten erhalten bleiben. Schaltungstechnisch wird sichergestellt, daß beim Abschalten der Betriebsspannung 5P des Kleincomputers der Datenerhalt gewährleistet wird, indem im Abschaltmoment die SRAM-Eingänge $\overline{CS1}$–$\overline{CS4}$ und $\overline{WE}$ inaktiv werden. Eingangs- bzw. ausgangsseitig ist der SRAM mittels Pull-up- bzw. Pull-down-Widerständen (130 KΩ) versehen, um den signalmäßig inaktiven Zustand zu erzwingen. Die Freigabe des SRAM erfolgt etwa 2 s nach Anliegen der Betriebsspannung 5P.

Die Entkopplung des SRAM vom Datenbus des KC erfolgt durch den Schaltkreis DS 8286 D. Der Betrieb des Erweiterungsmoduls erfordert keinerlei Eingriffe in den Kleincomputer.

Für die Adreßdecodierung wurden zwei Varianten entwickelt, die durch eine unterschiedliche Bestückung der Leiterplatte realisiert werden. Entsprechend den Forderungen des ersten Einsatzfalles können mittels 2 DIL-Schaltern folgende Speicherbereiche eingestellt werden:
- 4000H – 4FFFH
- 6800H – 77FFH
- 8000H – 8FFFH
- A800H – B7FFH

Eine zweite Variante ermöglicht die Adressierung von jeweils zwei 2-KByte-Blöcken je Modul innerhalb eines 16-K-Adreßbereiches. Bei der Festlegung der Adreßbereiche ist unbedingt die Speicheraufteilung des KC einschließlich Erweiterungsmodule (RAM, ROM, BASIC) zu beachten, um Doppelbelegungen bestimmter Bereiche zu vermeiden.

Bildet der SRAM-4-KByte-Erweiterungsmodul das Ende des RAM-Bereiches, dann wird das END OF RAM in diesen eingetragen.

Die Betriebsspannung 5P gelangt sofort nach Einschalten des Kleincomputers an alle bipolaren Schaltkreise des Moduls.

Gleichzeitig werden die unipolaren Schaltkreise (V 4066, SRAM 85 631) von der Schlafspannung auf 5P umgeschaltet. Zum Feststellen des Spannungszustandes der NK-Akkus dient eine Triggerschaltung. Diese signalisiert mittels rotleuchtender LED („AK") eine Gesamtspannung der Akkus kleiner 3V, d. h. die Schlafspannung am SRAM vor Einschalten des Kleincomputers war kleiner 2 V und Datenverlust ist möglich. Nach etwa 2 s werden die Akkus auf Erhaltungsladung umgeschaltet. Schaltungstechnisch ist garantiert, daß keine Überladung der Knopfzellen bei ununterbrochenem Betrieb erfolgt.

Mit der Umschaltung der Akkus auf Erhaltungsladung werden die mittels CMOS-Analogschalters unterbrochenen Speicherzugriffssignale CS1 ... 4 und WE durchgeschaltet. Damit ist ein Zugriff auf den SRAM-Bereich möglich. Diese Bereitschaft wird mittels grüner LED („ON") signalisiert.

Das Laden der Akkus kann bei gezogenem Modul über einen speziellen Steckkontakt, der nach Abnahme des oberen Gehäusedeckels zugänglich ist, erfolgen. Während der Ladezeit muß eine Drahtbrücke geöffnet werden. Für die Aufladung des NC-Akkus gelten die vom Hersteller angegebenen Vorschriften.

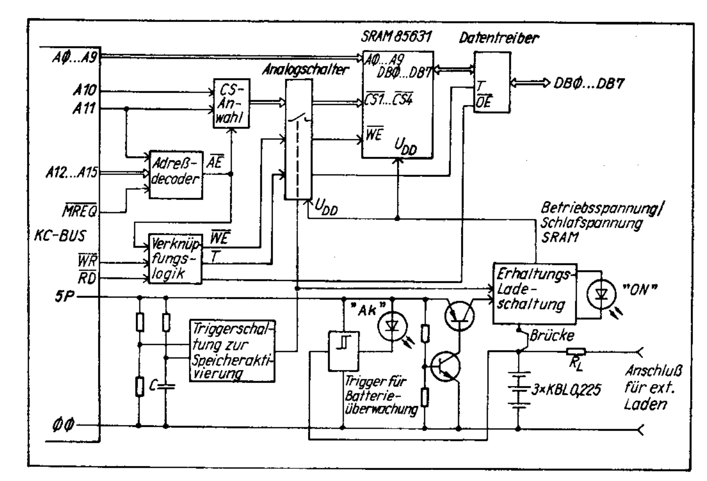


Bild 1 **Blockschaltbild**

Anwendungsmöglichkeiten

Der SRAM wurde erstmals für die Datenerfassung in Fertigungsbereichen eingesetzt. Die zu erhaltenden Daten wurden in komprimierter Form in den SRAM eingetragen und gingen nach plötzlichem Netzausfall, z. B. bei Betätigung der Nottaste, nicht verloren. Die gespeicherten Dateien konnten bei voll geladenen NC-Akkumulatoren nach einer Lagerung von mehreren Wochen noch fehlerfrei aus dem SRAM ausgelesen werden.

Der SRAM-4-KByte-Erweiterungsmodul kann anstelle von dRAM- und ROM-Modulen unter Beachtung der geringen Speicherkapazität eingesetzt werden. Bei vorhandenen freien Steckplätzen können auch mehrere SRAM-Module betrieben werden.

Nachnutzungsfähige Unterlagen zum Aufbau des SRAM-Moduls sind im Betrieb vorhanden und können über das BfN angefordert werden. Eine 16-KByte-Variante des SRAM-Moduls befindet sich gegenwärtig in Vorbereitung.

Literatur

/1/ Technisches Datenblatt „SRAM in CMOS-Technik". VEB Keramische Werke Hermsdorf

/2/ Technisches Datenblatt „Gasdichte Nickel-Kadmium-Akkumulatoren". VEB Grubenlampenwerke Zwickau

/3/ Keller, G.; Kleinmichel, G.: Der neue Kleincomputer im Überblick. Mikroprozessortechnik 1 (1987) 1, S. 22

/4/ Betriebsdokumentation Mikrorechnersystem K 1520. VEB Robotron-Elektronik Zella-Mehlis 1983

Menütechnik auch in SuperCalc

Ein grundlegender, bei der Erarbeitung von Programmen für Personalcomputer zu berücksichtigender Aspekt ist die leichte und überschaubare Bedienerführung im Bildschirmdialog. Das SC-Tableau allein trägt dem kaum Rechnung.

Mittels Textverarbeitungssystem *Textprozessor* geschriebene SC-Befehlsfolgen (XQT.Dateien) sind es, welche die Arbeit mit dem Kalkulationsprogramm überhaupt erst effektiv werden lassen. Ihr Aufruf erfolgt mittels des Befehls /X (execute).

Nachfolgend wird eine leicht handhabbare Methode der Bedienerführung über Bildschirm-Menü für SC-Programme vorgestellt.

Das Grundprinzip ist sehr einfach. Durch Veränderung im Betriebssystem SCP (Driver-Installation), die mit einem Diensthilfsprogramm durchgeführt wird, veranlaßt man, daß die einzelnen Programmzweige (Pk, k = 1, ... n)
P1.XQT
P2.XQT

Pn.XQT
statt durch die Befehle /XPk.XQT mittels vorher definierter Tasten (Tk, k = 1, ..., n) (zweckmäßig Funktionstasten)
gestartet werden.

Die Information darüber, welche Taste Tk welche Programmfunktion Pk auslöst, erhält der Bediener über den Bildschirm in Form eines Menüs angezeigt (Beispiel siehe Bild 1).

Driver-Installation

Um die Tasten Tk zu definieren, mit denen der Start der Programmzweige Pk ausgelöst wird, bedient man sich des Dienstprogrammes INSTSCP, darin des Programmzweiges 'DRIVER INSTALLATION'.

Beispiel:
Driver Installation für
Programmzweig P10.XQT
auf Fkt.Taste F10
/XP10.XQT entspricht im
ISO-7-Bit-Code: 2F,58,31,30
F10 entspricht in SCP-Codierung (hex): 83

COLD-START-Installation

Wie bereits vorn gesagt, soll ein dem Programm vorangestelltes Menübild dem Bediener die Abarbeitung des Programms erleichtern. Es gibt zugleich einen Überblick über die Leistungsmöglichkeiten des Programms. Erstellt wird dieses Menübild als Datei vom Typ .cal. Es kann demzufolge unmittelbar mit /L(oad) auf den Bildschirm gerufen werden. Als bedienerfreundliche Variante bietet sich an, den Kaltstart des Personalcomputers mit dem Laden des Programms *SC* und des Menübildes in einem gekoppelten Vorgang zu vollziehen. Das bedeutet, daß nach Einlegen der Diskette und Auslösen des Kaltstarts nicht nur das Betriebssystem, sondern auch *SC* und das Eingangsmenü (z. B. menue.cal) geladen werden.

Das geschieht, indem man
1. eine XQT.Datei (z. B. prog. XQT) mit Hilfe des Textverarbeitungsprogramms (TP) schreibt, welche
/ZY Löschen Bildschirm
/Lmenue.cal,A Laden Menü
/GB Ausschalten Spalten-/Zeilenbeschriftung
beinhaltet;
2. mittels des Dienstprogramms INSTSCP eine COLD-START-Installation für das KP-Programm (hier: prog.XQT) durchführt
hier: SC prog

Zwei Bemerkungen zum Abschluß
- In den Tableaus, die auf dem Bildschirm erscheinen, müssen natürlich ebenfalls Bedienerhinweise zur Funktion der Festtasten gemacht werden (mindestens: Rücksprung zum Menübild).
- Beim Kopieren von Programmen mit COLD-START-INSTALLATION und Programmen mit installierten Funktionstasten (DRIVER-INSTALLATION) ist zu beachten, daß außer den Dateien der Typen .XQT und .CAL des Programms auch das modifizierte SCP-Betriebssystem übernommen wird.

E. Poerner, B. Schleicher

```
*******************
 MENUE
*******************

Das Programm realisiert bei Betaetigen der
angegebenen Funktionstasten folgende Funktionen:

  F1        Eingabe von Stammdaten
  F2        Eingabe von variablen Daten
  F3        Verarbeitung der variablen Daten
  F4        Druck der Ergebnisliste
  F5        Anzeige Ergebnis auf Bildschirm
  F6        Austritt aus dem Programm
  F7        Austritt aus KP

****************************************************
****************************************************
```

Bild 1 Menübeispiel

```
<--¦     9D        F1   D1
                   F2   D2
--->     86        F3   D3
                   F4   D4
¦<--     87        F5   CF
                   F6   A0
<---     88        F7   A1
                   F8   A2
 ↖                 F9   A3
   ¦     8C        F10  83
                   F11  C1
   ^               F12  C0
   ¦     8B        F13  C2
                   F14  CD
--->¦    89

   ¦     8A
   v
```

Bild 2 Hexadezimalcode ausgewählter Tasten

Vorsicht, Satire!

Von der Vertreibung der GOTO-Kobolde

Es war einmal vor langer, langer Zeit, als viele Programme noch arg verwinkelt aussahen wie die Gäßchen alter Kleinstädte; da steckten sie oftmals voller tückischer GOTO-Kobolde. Wo diese nur konnten, brachten sie mit übermütigen, wilden Sprüngen alles gründlich durcheinander, und den Programmierern, die sich darob verzweifelt die Haare rauften, zeigten sie eine lange Nase und grinsten diese dreist und unverschämt an.
„So kann das nicht weitergehen", beschlossen eines Tages die Programmierpäpste, und sie riefen die Exorzisten Boehm, Jacopini und Weizenbaum zu Hilfe, auf daß sie die Bösewichte vertrieben. Alsbald hatten jene dann auch ein Mittel dafür parat, welches sie „Strukturierung" nannten. „Wenn ihr eure Programme künftig nur aus glatten Strukturblöcken erbaut, dann gibt es darin keine Ritze mehr, in der so ein Übeltäter noch Platz fände!" verkündeten sie ihr Rezept. Als das die geplagten Programmierer vernahmen, waren sie überaus froh; alle jubelten laut über den trefflichen, rettenden Einfall, priesen jene, die ihn hatten, über alle Maßen und verfuhren danach.

Die schlauen GOTO-Kobolde jedoch schlugen ihren Austreibern ein Schnippchen. Flugs nahmen sie die unterschiedlichsten Gestalten an und schlüpften verkleidet sogleich in die Strukturblöcke, kaum daß diese nur erdacht waren. So erschienen die einen fortan als WEND, WHILEND oder ENDWHILE, die anderen als DOEND, IFEND, ENDDO oder ENDIF, auch gaben sich manche zur Tarnung gar absonderliche Namen wie ADMIT, EXIT, QUIT, NEXT, LOOP, EXCEPTION oder RAISE. Die meisten aber nannten sich schlicht und nichtssagend nur END; ja, den boshaftesten Schlitzohren unter ihnen gelang sogar das Kunststück, sich in ein unscheinbares Semikolon zu verwandeln.

Jetzt erkennt keiner mehr ihr hinterhältiges Wesen, weil sie alle ebenso ehrbar und bieder aussehen wie jede andere Anweisung. Auch verraten sie sich nicht mehr durch Grinsen, sondern blicken einer wie der andere mit stets gleichmütiger, ausdrucksloser Pokermiene gelangweilt ins Leere, weil eine Maske ja keine Miene verzieht, wie man weiß. Und ihre Sprünge machen sie nur heimlich im Dunkeln, so daß keiner sie bemerkt. Aber auch heute noch treiben sie – gut getarnt und unerkannt – ärgeren Schabernack als zuvor, besonders wenn sie sich an den Programmenden zu langen Ketten versammeln. Und wenn dann die ahnungslosen Programmierer ob der Verwirrung oder gar falscher Abläufe in den Programmen wiederum fluchen oder sich die Haare raufen, lachen jene sich stillvergnügt und heimlich ins Fäustchen.

Und nur einige ältere Programmierer gedenken dabei wehmütig der guten alten Zeiten, da man die grinsenden GOTO-Kobolde alle noch leicht erkennen, verfolgen und zur Strecke bringen konnte, und mancher fragt sich nunmehr insgeheim zweifelnd, ob die Exorzisten seinerzeit wirklich den besten Einfall hatten. Aber welcher aufgeklärte Software-Designer von heute glaubt noch an das Märchen von der Kobold-Vertreibung. Die Tatsache, daß er in den modernen, wohlstrukturierten Programmen jederzeit voll durchblickt und daß dort stets alles mit rechten Dingen zugeht, hat ihn völlig davon überzeugt, daß es GOTO-Kobolde (Wirth sei Dank!) in Wirklichkeit längst schon nicht mehr gibt (o d e r ?).

D. Holz

Direktzugriff auf REDABAS-Dateien unter TURBO-PASCAL

Das Programmodul PARE ermöglicht ein direktes Bearbeiten von REDABAS-Dateien unter TURBO-PASCAL ohne Dateikonvertierungen. Die Anwendung des Programms ist sehr einfach und erlaubt auch den Dateityp DBF aus dBASE-II-Dateien. Lauffähig ist PARE unter SCP auf A 5120/30, PC 1715 sowie auf A 7100.
Kenndaten des Programms:
– PARE liegt als Quelltext vor und kann mittels INCLUDE-Anweisung am Anfang eines Anwenderprogramms eingebunden werden.
– Der Quelltext beansprucht einen Speicherplatz von ca. 10 KByte, im kompilierten Anwenderprogramm werden zusätzlich max. 6 KByte belegt.
– Der modulare Aufbau von PARE gewährleistet eine optimale Anpassung an Anwenderprogramme; nicht benötigte Prozeduren/Funktionen können zur Speicherplatzeinsparung im Quelltext gestrichen werden.
– Feldvariablen der verwendeten REDABAS-Dateien brauchen vom Anwender nicht vereinbart werden, da das Eröffnen einer REDABAS-Datei automatisch das Erfassen ihrer Struktur bewirkt. Name und Typ der Feldvariablen müssen an die entsprechenden Prozeduren/Funktionen übergeben werden.
– Befehlssatz: Datei eröffnen, schließen; Satz einstellen, suchen, anzeigen, anhängen; Wert lesen, schreiben
– Es werden die Standardtypen STRING, CHAR, REAL und INTEGER genutzt.
– Mit dem Zusatzprogramm STRU können Strukturen von REDABAS-Dateien direkt, also ohne Zugriff über das Datenbanksystem, aufgelistet werden.

Institut für Sozialistische Betriebswirtschaft, Bereich Stadtroda/Forschungsgruppe Anwendersoftware, August-Bebel-Straße 3, Stadtroda, 6540; Tel. 2 10 13/2 10 91

Dr. Ahnert

Lesen und Schreiben von ESER-Magnetbandfiles an SKR-Rechnern

Der Ausgangspunkt zur Schaffung des Programms ESTAPE war, daß für SKR-Rechner wie CM-4, CM-52, CM-1420, K 1630, I102 und für Anlagen des Typs PDP-11 bereits diverse ESER-Magnetband-Konvertierungsprogramme vorlagen, diese sich aber zumeist mit einer Konvertierung von Zeichenketten begnügten, wobei auch andere Probleme mit der praktischen Handhabbarkeit auftraten.
ESTAPE ermöglicht die Konvertierung von MB-Files im ESER-Format in sequentielle SKR-Plattenfiles (und umgekehrt) unter zu OS-RW (ab V2.8) und RSX-11 M (ab V3.2) vergleichbaren Betriebssystemen.
Auf beiden Seiten sind Records fester oder undefinierter Länge zulässig, Records fester Länge auf MB können geblockt sein. Das MB kann im DOS- oder OS-Format bzw. als Nolabel-Band verarbeitet werden.
Insbesondere konvertiert ESTAPE Datenfelder aus beliebigen Internformaten des ESER in entsprechende SKR-Internformate und umgekehrt.
ESTAPE verfügt darüber hinaus über entsprechenden Komfort der Magnetbandarbeit, wie Möglichkeit zur Initialisierung und Labelprüfung, absolute und relative MB-Positionierung vor und nach dem Transfer, Arbeit mit Filenamen, wobei diese auch (unter Einschränkungen) in beiden Richtungen automatisch übernommen werden können. Beim MB-Schreiben werden alle notwendigen Standardsätze (auch HDR2 usw.) vollständig erzeugt. Als Schreibdichte sind 800 (Standard) und 1600 bpi zulässig.
ESTAPE ist sowohl mit direkter, der UNIX-Schalterkonvention angenäherter Kommandoeingabe oder aber über spezielle Kommandofiles arbeitsfähig, wodurch Routineprojekte auch im Stapelbetrieb abgearbeitet werden können.
Durch Nutzung der Möglichkeiten der Programmiersprache C arbeitet ESTAPE weitestgehend mit dynamischer Speicherverwaltung, was sich günstig auf den Mehrnutzerbetrieb bei kleinem Hauptspeicher (128 K Worte) auswirkt.
Die dargestellten Leistungen von ESTAPE sind Inhalt der aktuellen Version 1.5, die sich an unserer Einrichtung sowie bei einer größeren Anzahl von Nachnutzern in der praktischen Arbeit bewährt hat. Eine Portierung von ESTAPE an die neuen Rechner der 32-Bit-Technik dürfte ohne große Probleme möglich sein.
Allen Nachnutzern werden eine lauffähige Task, ein Generierungskit auf Objektmodulbasis sowie die ausführliche Dokumentation übergeben.

ORGREB – Institut für Kraftwerke, Bereich Dresden, Abt. 2720, Mügelner Str. 27, Dresden, 8036; Tel. 4849491 oder -4 95

Benedikt

Universelles Editierprogramm

Zur Nachnutzung bieten wir ein Programm zur Ver- und Bearbeitung von Daten im Lochkartenformat an. Das Programm ist lauffähig auf 8- und 16-Bit-Mikrorechnern unter den Betriebssystemen SCP, CP/A und MS-DOS und beinhaltet folgende Funktionen:
– Neuerstellen einer Lochkartendatei
– Erweitern
– Einfügen
– Korrigieren
– Anzeigen
– Löschen
– Drucken von Sätzen (Lochkarten) der Datei
– Konvertieren zum Wandeln der Lochkartendatei in eine sequentielle Datei und umgekehrt
– Steuercodes generieren zur Anpassung an unterschiedliche Rechner.

Das Programm ist in TURBO-PASCAL geschrieben und benötigt einen Speicherplatz von etwa 40 KByte.

EVDR, BT RAT, TK SW, Frankfurter Allee 216, Berlin, 1130; Tel. 4942693

Schubert

Farbgrafik für den PC 1715

Diese Hard- und Softwarelösung dient der Darstellung von farbigen Grafiken auf einem handelsüblichen Farbfernsehgerät mit RGB-Anschluß. Bei einer Auflösung von 512×256 Pixeln sind 16 aus 4096 Farben darstellbar. Der Anschluß an den PC erfolgt wahlweise über eine nachzurüstende Parallelschnittstelle oder die standardmäßig vorhandene V.24-Schnittstelle. Durch Einsatz eines eigenen Mikrorechners verfügt die Grafik über einen umfangreichen Befehlssatz und entlastet den PC von zeitintensiven Grafikroutinen. Anschließbar sind ein Nadeldrucker zur Hardcopy sowie eine Maus zur interaktiven Bildschirmarbeit. Durch eine Softwarelösung steht dem Anwender ein Grafikbefehlssatz in TURBO-PASCAL zur Verfügung.

Ingenieurhochschule Berlin, Prorektorat für Naturwissenschaft und Technik, Marktstr. 9, Berlin, 1134; Tel. 2071064

Siegmund

Grafik mit BASIC am AC A 7100

Im VEB Elmo Thurm wurde eine Grafikschnittstelle entwickelt, die es ermöglicht, mit der Programmiersprache BASIC die Grafikfähigkeit des A 7100 zu nutzen. Die Schnittstelle stellt eine Verbindung zwischen dem BASIC-Interpreter und der Grafikerweiterung SCP-GX des Betriebssystems SCP 1700 her. Der BASIC-Anwender kann auf einfache Art und Weise die wichtigsten Grafikfunktionen des SCP-GX nutzen, ohne spezielle Kenntnisse über dessen Aufbau und Wirkungsweise zu besitzen.
Voraussetzung für die Anwendung dieser Grafikschnittstelle sind die Grafikvariante des A 7100 und die entsprechende Systemsoftware unter dem Betriebssystem SCP.
Für Interessenten wird zur Nachnutzung neben dem Programm eine Anwendungsbeschreibung mit Beispielen angeboten.

VEB Elektromotorenwerke Thurm, Abteilung TN, Ernst-Thälmann-Str. 86, Thurm, 9527; Tel. Zwickau 827219

Weigel

PullDown-Menü für den PC 1715

Die PullDown-Menü-Technik bietet eine übersichtliche, optisch anspruchsvolle Lösung für den Nutzer und eine wesentliche Rationalisierung für den Programmierer.
Der konkrete Menüaufbau ist im aufrufenden Programm in Form eines speziellen Feldes zu definieren. Das Menü kann bis zu 6 Hauptpunkte (jeder max. 12 Zeichen) besitzen, die jeweils 10 Unterpunkte (jeder max. 22 Zeichen) haben dürfen.
Durch Aufruf des Assembler-UP PDMENUE kann das Menü an beliebigen Stellen des Programms „über" dem bestehenden Bild eingeblendet werden. Der Benutzer wird dadurch zur Auswahl aufgefordert, die er mit Hilfe der Cursortasten vornimmt. Vor jedem Menüaufruf kann eine beliebige Zahl von Menüpunkten inaktiviert werden. Diese Punkte werden mit dem kompletten Menü angezeigt, sind aber für die Auswahl gesperrt.
Nach erfolgter Auswahl übernimmt PDMENUE die Wiederherstellung des Bildschirminhaltes und liefert dem aufrufenden Programm Angaben zum ausgewählten Punkt. Vom Bediener kann nur ein vorhandener und aktiver Punkt des Menüs gewählt werden. Das Unterprogramm PDMENUE wurde entwickelt für den PC 1715 mit einem Monitor BAB2 (24*80 Zeichen). Zur Verfügung stand das Betriebssystem SCP 0.5 mit zusätzlich installierten Cursortasten.
Das Programm ist auch lauffähig unter Betriebssystemen (z. B. CP/A), die die gleiche Anfangsadresse des Bildwiederholspeichers (=F800H) sowie gleiche Codierung für Bildschirmsteuerzeichen und Cursortasten besitzen.
Bei Zusendung einer 5,25"-Diskette können interessierte Nachnutzer kostenlos ein Demonstrationsbeispiel erhalten.

Hochschule für Musik „Carl Maria von Weber", Studio für elektronische Klangerzeugung, Blochmannstr. 2–4, Dresden, 8010

Lange

Handbuch „Softwareentwicklungsarbeitsplatz 16-Bit-PC"

Das Anliegen des Handbuches ist die Unterstützung des Softwareentwicklungsprozesses auf der Grundlage des Betriebssystems DCP. Mit dem Handbuch soll eine technologisch effektive Anwendung der Möglichkeiten des Betriebssystems und der vorhandenen unterstützenden Software vorgegeben werden. Die dazu notwendige Nutzeroberfläche wird in Form von Kommandoprozeduren bereitgestellt.
Das Handbuch beinhaltet folgende Hauptthemen:
– Hardwarebesonderheiten der einzelnen PCs
– Konventionen
– Vor und Nachbereitung einer PC-Sitzung
– DCP-Werkzeuge
– Dateiarbeit
– Sprachunterstützung
– Unterstützung komplexer Softwaresysteme.

Innerhalb der Hauptthemen erfolgt die Beschreibung der notwendigen Werkzeuge und der dazugehörigen Kommandoprozeduren mit Beispielen.

VEB DVZ Halle, Abteilung FP, Block 081, Halle-Neustadt, 4090; Tel. 616404

Gruhl

TERMIN

Fachtagung „CAD-Systeme für Elektroenergieanlagen"

WER? Arbeitsausschuß „Rechnergestützte technische Vorbereitung von Elektroanlagen" in der Bezirksfachsektion Elektrotechnik/Elektronik beim Bezirksvorstand Leipzig der Kammer der Technik
WANN? 13. September 1988
WO? Leipzig
WIE? Interessenten wenden sich formlos, schriftlich oder telefonisch an KDT, BV Leipzig, SB Wissenschaft und Technik, PF 40, Leipzig 7010

Schwendler



Einsatz von Flachbildschirmen

Nach einer Studie der Firma Frost & Sullivan soll sich das Einsatzgebiet von Flachbildschirmen, die bisher vornehmlich in Fernsehgeräten zum Einsatz kamen, zur Autoindustrie und die Computertechnik verlagern. In der Computertechnik kommen Flachbildschirme hauptsächlich in tragbaren Computern zum Einsatz, während sie in Kraftfahrzeugen als Armaturenbrett eingesetzt werden sollen.
Bezüglich der Technologien soll die bedeutendste Entwicklung die sogenannte Super-Twist-Variante von Flüssigkristall-Anzeigen sein. Sie verbessern den Betrachtungswinkel beträchtlich. Sie sollen unter den acht Anzeigetechnologien die höchsten Zuwachsraten aufweisen. Plasmabildschirme erreichen nur etwa halb so große Zuwachsraten und LCDs nur ein Drittel. Bis zum Jahre 1992 sollen LCDs jedoch die marktbeherrschende Technologie bleiben.

Aus Elektronik Entwicklung. – Wiesbaden 22 (1987) 12. – S. 14

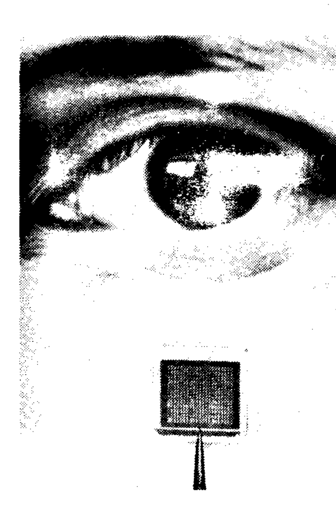
Expertensystem zur Bestimmung der Rechnerkonfiguration

Bereits seit einigen Jahren befassen sich verschiedene Institutionen in der ČSSR mit der Schaffung von Expertensystemen. Auf der Konferenz über Künstliche Intelligenz in Prag im Jahre 1986 wurde über die Systeme CODEX, EQVANT, SAK, CKD, VEX, KONSULT und FEL-EXPERT berichtet. Auf der Grundlage des letzteren wurde von einem interdisziplinären Kollektiv ein Expertensystem zur Bestimmung der Kapazitäten der Rechentechnik geschaffen. Die Wissensbasis besteht aus Informationen über die verfügbare Rechentechnik. Das Expertensystem ermöglicht das stufenweise Sichten der Anforderungen, Kenntnisse und Voraussetzungen der Nutzer. Es schlägt die minimale Ausstattung an Rechentechnik vor. Gegenwärtig ist die Wissensbasis auf die Auswahl von Kleinrechentechnik einschließlich der an Zentralrechner angeschlossenen Terminals ausgelegt. In einer weiteren Ausbaustufe soll die Konfiguration von Zentralrechnern bestimmt werden können.
Das Kriterium „Bedürfnisse des Nutzers“ schließt die Prüfung des Anwendungsgebietes, die Art und Weise der vorgesehenen Arbeit mit dem System und die Anforderungen an Hauptspeicher, periphere Speicher und E/A-Geräte ein. Bei dem Kriterium „Voraussetzungen des Nutzers“ werden die finanziellen Mittel, die räumlichen und technischen Voraussetzungen, die Personalkapazitäten und die Erfahrungen bei der bisherigen Nutzung von Rechentechnik berücksichtigt. Die ausgewählten Alternativen werden jeweils mit den Voraussetzungen des Nutzers verglichen, bis eine widerspruchsfreie Lösung vorgeschlagen ist.
Der Wert des Expertensystems besteht darin, daß auf den Erfahrungen von Experten beruhende Konfigurationen von Rechentechnik zum Kauf vorgeschlagen werden, die für die Lösung der anstehenden Probleme ausreichend sind. Gegenwärtig wird eingeschätzt, daß 15 % der im tschechischen Teil der ČSSR installierten Rechner überdimensioniert sind. Geht man von einem Installationswert von 2423 Mio Kčs (1985) aus, so ist es durch Optimierung der Konfiguration möglich, die Anschaffungskosten um etwa 363 Mio Kčs zu senken, davon 218 Mio Kčs Deviseneinsparungen.
Das Expertensystem ist auf dem Rechner ADT 4700 mit dem Betriebssystem RTE und DOS 4 implementiert. Es soll jedoch auch auf den Rechnern PDP 11, CM 3, CM 4, JPR 12 und IBM-PC lauffähig sein.

Quelle: mechanizace a automatizace administrativy (1987) 12. – S. 462

Wi

Transistor mit 0,1 μm Gatelänge



Forschern der Cornell University, Ithaka, N.Y., und von Siemens ist es gelungen, einen Galliumarsenid-Transistor (GaAs) für deutlich mehr als 100 Gigahertz herzustellen. Dieser „Modfet“ (modulation-doped field-effect transistor) arbeitet mit einem Sandwichchip, dessen Schichten nur wenige Hundertstel Mikron „dick“ sind und in einem von den Forschern entwickelten Spezialverfahren mit Silizium sehr exakt dotiert werden. Als Transitfrequenz (unity current gain cut-off frequency) wurden bisher 113 GHz gemessen.
Der annähernd quadratische GaAs-Wafer im Bild integriert 250 Modfet.
Insbesondere Experten für Satelliten, Raumfahrt und Radar interessieren sich für diesen Transistor bisher unerreichter Schnelligkeit.

International Solid State Circuits Conference

Während bei der ISSCC'87 das Schwergewicht sowohl hinsichtlich der Anzahl der Vorträge als auch ihrer Bedeutung bei den Speicherschaltkreisen lag, zeigt das Programm der ISSCC'88 ein ausgewogenes Verhältnis zwischen Halbkundenschaltkreisen, Prozessoren, analogen Schaltkreisen und RAM-, ROM- und nichtflüchtigen Speicherschaltkreisen. Höhepunkte bei den Speichern sind ein 16-MBit-DRAM von Matsushita und ein 4-MBit-DRAM von Siemens mit Zugriffszeiten von 60 ns, zwei SRAMs in BICMOS mit 256 KBit Speicherkapazität von Hitachi und Fairchild sowie ein ferroelektrischer statischer RAM von RAMTRON mit einer Lebensdauer von max. 10^8 Umprogrammierungszyklen (siehe Tafel 1).

Tafel 1

	Speicherkapazität	Zugriffszeit	Technologie/ Struktur	Speicherzelle Größe	Firma
DRAM	16 MBit	60 ns	CMOS 0,5 μm	3,3 μm²	Matsushita
DRAM	4 MBit	60 ns	CMOS 0,9 μm	10,58 μm²	Siemens
SRAM	256 KBit	8 ns	BICMOS 1 μm	66,3 μm²	Hitachi
SRAM	256 KBit	12 ns	BICMOS	96,5 μm²	Fairchild

Auf dem Gebiet der Halbkundenschaltungen wird von Actel Corp. eine neue Technologie für anwenderkonfigurierbare, elektrisch programmierbare Logikschaltungen vorgestellt. Sie verwendet eine sogenannte „Antifuse“-Architektur für die Programmierung von horizontalen und vertikalen Elementen. Dies bedeutet eine Abweichung sowohl von den traditionellen CMOS-PLDs, die eine gespeicherte Ladung zur Programmierung einer Zelle einsetzen, als auch von den bipolaren PLDs, die eine Schmelzsicherung einsetzen. Das Ergebnis ist eine beträchtlich erhöhte Packungsdichte. An der Spitze der Prozessoren steht ein 32-Bit-Mikroprozessor von Texas Instruments mit RISC-Architektur auf der Basis von Galliumarsenid. Der Mikroprozessor ist ausgestattet mit einem 16 × 32-Bit-Register, einer 32-Bit-ALU und sechsstufiger Befehls-Pipelineverarbeitung. General Electric berichtet über einen 32-Bit-Prozessor in CMOS mit einer Taktfrequenz von 40 MHz und einer Zugriffszeit von 25 ns.
Auf dem Gebiet der Analogschaltungen stellt IBM ein automatisches Plazierungs- und Routing-System für gemischte Analog- und Digitalschaltungen vor, das einen Chip-Ausnutzungsgrad von 75 % erreichen soll. Große Aufmerksamkeit wird fortgeschrittenen Entwürfen von A/D- und D/A-Wandlern gewidmet, vorgestellt von den Firmen Analog Devices, Hitachi, NEC, Philips und AT & T.

Quelle: Electronics. – New York 60 (1987) 25. – S. 103–104

Fa

Eine neue schnelle Bipolarlogik mit niedriger Verlustleistung

Forscher des IBM Forschungslabors von Böblingen in der BRD haben eine neue Bipolarlogik entwickelt, deren durchschnittliche Verlustleistung bei Beibehaltung hoher Geschwindigkeit auf das Niveau von CMOS-Schaltungen reduziert worden ist: 50 Mikrowatt Verlustleistung und eine Laufzeit von 800 ps wurden in experimentellen Schaltungen erreicht.
Die neuen komplementären Transistor-Logik-Schaltungen verwenden Minoritätsladungsträger, um die Ströme zum Schalten der Ausgangstransistoren zu erzeugen. Daher stammt auch der Name der neuen Logik: CBL-Charge Buffered Logik. Da im Ruhezustand nur niedrige Gleichströme fließen, erreicht die durchschnittliche Verlustleistung CMOS-Werte. Die neue Logik verhält sich zur Bipolarlogik wie CMOS zu MOS: Beide sind komplementäre Schaltungen, wobei CBL pnp- und npn-Transistoren einsetzt und MOS sich auf p- und n-Kanal-Transistoren stützt.
Um die o.g. 800 ps und 50 Mikrowatt zu erreichen, setzt das IBM-Forschungsteam eine 2,2-μm-Struktur und 200-MHz-Schalttransistoren ein. Computerberechnungen haben jedoch gezeigt, daß bei Einsatz einer 1,2-μm-Struktur und 1,5-GHz-pnp-Transistoren 300 ps und weniger als 10 Mikrowatt erreicht werden können, bei einer 1-μm-Struktur, Grabenisolation und anderen neuen Prozeßverfahren sogar 100 ps.
Bis jetzt hat man sich international noch kaum mit bipolarer komplementärer Transistorlogik beschäftigt, da sich der stromgesteuerte Bipolartransistor im Gegensatz zum spannungsgesteuerten MOS-Transistor schlecht für komplementäre Logikanwendungen eignet: Nach dem Schalten fließt der Strom weiter, was eine ziemlich hohe statische Verlustleistung verursacht. Die Lösung scheint eine CMOS-ähnliche bipolare komplementäre Transistor-Logik zu sein, in der Bipolartransistoren MOS-Typen ersetzen. Die Implementierung einer solchen Logik mit Bipolarstrukturen erweist sich jedoch selbst mit fortgeschrittensten Prozeßtechniken als schwierig. Darüber hinaus begrenzen die relativ großen Sättigungs-Konstanten der gegenwärtig untersuchten Transistorstrukturen die Geschwindigkeit ernsthaft.
Diese Nachteile führten schließlich dazu, daß man sich den ladungsgesteuerten Logikschaltungen zuwandte. Während der dreijährigen Entwicklungszeit der CBL-Logik bestand die Schwierigkeit darin, einen Weg zu finden, eine große gesteuerte Ladung in die Basis eines Transistors einzubringen, ohne große Datenströme zu erzeugen, was konventionelle Bipolartransistoren fordern. Dadurch, daß diese Ströme so niedrig wie möglich gehalten wurden, konnte die Standby-Leistung beträchtlich gesenkt werden.

Quelle: electronics. – New York 60 (1987) 25. – S. 42, 43

Fa



Informatikliteratur aus dem VEB Verlag Technik

Die außerordentliche thematische Breite der Informatik kam besonders auf dem 4. Kongreß der Informatiker der DDR, INFO 88, zum Ausdruck. Die Vorträge wurden innerhalb der Fachsektionen theoretische Grundlagen, Computertechnik, Software, künstliche Intelligenz, komplexe Anwendungen, Aus- und Weiterbildung sowie Gesellschaft und Informatik gehalten. Diese thematische Gliederung soll dem Leser die Themenkomplexe veranschaulichen, wofür durch die Verlage der DDR entsprechende Literatur bereitzustellen ist. Um diesen Anforderungen gerecht zu werden, wird die Herausgabe von Literatur zu ausgewählten Schlüsseltechnologien zwischen den Verlagen abgestimmt. Der VEB Verlag Technik orientiert dabei vorrangig auf die Abdeckung des Informationsbedarfs für den Einsatz von Personalcomputern und CAD/CAM-Technik. Bereits seit einigen Jahren wird Literatur zu dieser Thematik vom Verlag herausgegeben. Die Bücher sind meist als praxisbezogene Fachbücher konzipiert, um die Anwendung der Rechner in der Industrie zu unterstützen. Im folgenden werden einige Bücher des Verlages, zum Teil geplante Literatur mit Arbeitstiteln, vorgestellt.

Software/Betriebssysteme

Die Bücher zur Software sind weitgehend rechnerunabhängig gestaltet.
Als Programmiersprachen finden in der DDR vorrangig BASIC, C, COBOL, FORTRAN, MODULA-2, PASCAL und deren Weiterentwicklungen Anwendung.
Die jetzige Titelplanung berücksichtigt diese Entwicklung mit den Büchern
- *Müller* „Programmieren mit BASIC"
- *Werner* „BASIC für Mikrorechner"
- *Schilling* „TURBO-BASIC"
- *Claußl/Fischer* „Programmieren mit C"
- *Kosciolowicz* „Programmieren mit FORTRAN 77"
- *Schiemangk* „Programmieren mit MODULA-2"
- *Hopfer* „Informationsverarbeitung mit PASCAL"
- *Paulin* „TURBO-PASCAL".

Für die in der DDR angebotenen Betriebssysteme SCP und DCP werden vom VEB Verlag Technik auch Bücher zu kompatiblen Betriebssystemen herausgegeben. Hier einige Titel aus unserem Programm:
- *Bernert/Burow/Hanisch* „CP/M in der Praxis"
- *Claßen/Oefler* „UNIX und C"
- *Polze* „UNIX-Werkzeuge zur Programmentwicklung"
- *Hübener* „MS-DOS".

Anwendersoftware

Durch den Einsatz von Personalcomputern in allen Bereichen der Volkswirtschaft sowie im persönlichen Bereich nimmt die Entwicklung von eigener Software immer mehr zu. Um hier anerkannten und neuen Methoden zum Durchbruch zu verhelfen, wurde zur Unterstützung des Softwareentwicklers das Buch von *Rothhardt* „Praxis der Softwareentwicklung" herausgegeben. Ergänzt wird diese Thematik durch *Lindner/Trautloft* „Grundlagen der problemorientierten Programmentwicklung", eine Anleitung zur systematischen Programmentwicklung.
Ein effektivitäts- und qualitätsbestimmender Bestandteil von Informations- und Steuerungssystemen in zahlreichen Anwendungsgebieten sind Datenbanksysteme. Durch den massiven Einsatz von Personalcomputern und leistungsfähigen Kleinrechnern nimmt die Anzahl der Datenbanken sprunghaft zu. Die folgenden Bücher des Verlages sollen den Datenbankspezialisten oder -anwendern eine Hilfe sein:
- *Grafik* „dBASE-Software für Datenbanken"
- *Trautloft/Lindner* „Datenbanken – Entwurf und Anwendung"
- *Schubert* „Datenbanksysteme".

Aufgrund der zu erwartenden technischen Entwicklung, z. B. automatisierte Fabrik (CIM) wird in den 90er Jahren der Bedarf an Literatur zu wissensbasierenden Systemen und KI-Programmsystemen stark zunehmen. Diese Entwicklung wird vom Verlag durch erste geplante Bücher berücksichtigt:
- *Helbig* „Künstliche Intelligenz und automatische Wissensverarbeitung"
- *Schwarz/Lunze* „Künstliche Intelligenz und intelligente Technik"
- *Baldeweg/Fiedler* „Expertensysteme in der technischen Diagnostik".

Hardware

Die derzeitige Entwicklung in der DDR umfaßt den verstärkten Einsatz von 16-Bit-Prozessoren, die Entwicklung von 32-Bit-Prozessoren sowie 1- und 4-Megabit-Speichern. Infolge der Typenvielfalt und der zentralisierten Produktion von Rechnern und Rechnerperipherie werden vom Verlag vorrangig allgemein einsetzbare Lösungen zur Schaltkreisentwicklung und Mikroprozessortechnik einschließlich deren Programmierung publiziert. Hier ebenfalls einige Titel:
- *Kieser/Bankel* „Einchipmikrorechner"
- *Franke* „Einführung in die Mikrorechentechnik"
- *Brennenstuhl* „Programmierung des 16-Bit-Mikroprozessorsystems U 8000"
- *Kieser* „16-Bit-Mikroprozessoren"
- *Bonitz* „Der 16-Bit-Mikroprozessor des ESER-PC"
- *Claßen/Wiesner* „Wissensspeicher 16-Bit-Mikroprozessor-Programmierung".

CAD/CAM, CIM

Die derzeitige Einsatzvorbereitung ist gekennzeichnet durch die Integration der betrieblichen Prozesse. Erforderlich sind hier u. a. intelligente Maschinensysteme, Expertensysteme, Rechnerverbundsysteme und automatische Transportsysteme. Durch unseren Verlag soll diese Entwicklung mit den Titeln
- *Pötschke/Lunze* „KI und Automatisierungstechnik"
- *Gottschalk/Wirth* „Bausteine der rechnerintegrierten Produktion – CIM"
- *Wirth* „Flexible Fertigungssysteme"
- *Gottschalk* „Rechnergestützte Produktionsplanung und -steuerung – PPS"
- *Krug* „Simulation für Ingenieure in CAD/CAM-Systemen"
- *Iwainsky* „Computergrafik in CAD/CAM-Prozessen"

unterstützt werden.
Anhand dieser ausgewählten Beispiele wurden dem Leser einige bereits erschienene bzw. geplante Buchtitel vorgestellt.
Die schnelle technische Entwicklung stellt an die Buchentwicklung hohe Anforderungen, und diese verläuft nicht immer ohne Probleme. Nicht alle Titel sind wegen begrenzter Kapazitäten immer im Handel verfügbar. Manche Titel erschienen erst, wenn Soft- oder Hardware bereits längere Zeit in der Volkswirtschaft genutzt wurden. Hier suchen wir nach Möglichkeiten zur Verkürzung der Herstellungszeiten und nehmen Einfluß auf die Manuskripterarbeitungszeiten. Ein gutes Beispiel ist die neue Reihe „Technische Informatik" unseres Verlages. Hier konnte durch enge Zusammenarbeit zwischen Herausgeber, Autor und Lektor die Manuskripterarbeitungszeit verkürzt werden. Die Herstellungszeiten verringerten sich durch die Erstellung reproduktionsfähiger Vorlagen als Computerausdruck vom Autor ebenfalls. Damit leisten wir einen Beitrag zur schnelleren Einführung der Schlüsseltechnologien.

Hartmut Heinrich

RFT

Berichte zur Nachrichtentechnik

Die Broschürenreihe aus dem Zentrum für Forschung und Technologie Nachrichtenelektronik

Folgende Bände sind noch lieferbar:

Band 2: Nachrichtenverkehrstheorie; Grundlagen und Berechnungsverfahren
Band 11: Analoge Übertragungstechnik auf Kabeln; Trägerfrequenzsysteme und -strecken
Band 14: Erzeugnisstandardisierung
Band 15: Vermittlungssystem ENSAD
Band 16: Digitale Nachrichtennetze
Band 17: Fernsprechendgeräte
Band 18: UKW-Verkehrsfunktechnik
Band 19: Fernsprechvermittlungstechnik
Band 20: Elektronische Vermittlungstechnik
Band 21: Digitale Nachrichtentechnik
Band 22: Ländliches Nachrichtenwesen
Band 24: Optimaldemulation
Band 25: Datenfernverarbeitung.

Umfang und Preis der Bände:
40 bis 160 Seiten; 6,50 M bis 14,40 M
Bestellungen sind zu richten an:

VEB Funkwerk Köpenick, Stammbetrieb im Kombinat Nachrichtenelektronik, Zentrum für Forschung und Technologie Nachrichtenelektronik (ZFTN), Abt. EA 3 (Versand), Edisonstraße 63, Berlin, 1160

80 Programme in TURBO-PASCAL

Von Michael Fothe. 120 Seiten. Sonderausgabe 1 der iir-Informatik „informationen, reporte", 1988. Herausgeber: Institut für Informatik und Rechentechnik der AdW der DDR, Rudower Chaussee 5, Berlin, 1199.

Diese Programmsammlung ist eine Fundgrube für jeden, der in TURBO-PASCAL programmiert und weitaus mehr als nur eine Sammlung von Programmbeispielen. Der Autor hat auf der Grundlage von Lehrgängen ein Material geschaffen, das die wesentlichen Sprachelemente von TURBO-PASCAL in ihrer Anwendung zeigt. Die gut dokumentierten und nach didaktischen Gesichtspunkten angeordneten 80 Programme sind sehr gut für einen Anfänger geeignet, sich in TURBO-PASCAL einzuarbeiten. Aber auch der Fachmann findet eine Reihe von Anregungen für professionelle Programme. Besonders geeignet ist das Material für Lehrgänge, vor allem, weil alle Programme lauffähig auf einer Diskette vorliegen. Die Lernenden können sofort die Wirkung der Befehle testen und die Programme selbständig erweitern und verfeinern.
Der Stil des Buches ist durch Übersichtlichkeit, Straffheit und gut aufeinander abgestimmte Darstellung des Stoffes gekennzeichnet. Das Verständnis der Programme wird durch die kurzen Erläuterungen, die jeweils vorangestellt sind, erleichtert. Ausgehend von Beispielen, die den Aufbau eines TURBO-Programmes beschreiben, werden die Anweisungsmöglichkeiten, Prozeduren, Funktionen, Felder, Zeichenketten, Records und Mengen beschrieben. Sieben Programme haben den Zeigertyp zum Inhalt. Es bleibt zu hoffen, daß es in der Zukunft eine Erweiterung der Sammlung geben wird, z. B. für die Turbo-Toolbox.

G. Hartmann

TERMIN

16. Fachtagung Mikrorechentechnik „CAD/CAM"

WER? Wissenschaftliche Sektion „Computer- und Mikroprozessortechnik" im Fachverband Elektrotechnik beim Präsidium der KDT gemeinsam mit dem Bezirksvorstand Leipzig der Kammer der Technik
WANN? 15. September 1988
WO? Leipzig
WAS? Übersicht über Grundlagen von CAD/CAM; Vorstellungen von praktischen Lösungen.
WIE? Interessenten wenden sich formlos, schriftlich oder telefonisch an KDT, BV Leipzig, SB Wissenschaft und Technik, PF 40, Leipzig 7010.

Schwendler



Leipziger Frühjahrsmesse 1988

Nachdem wir unseren Bericht von der Leipziger Frühjahrsmesse in MP 7/88 mit der Vorstellung der Bauelemente und Mini-/Mikrocomputer eröffneten, wollen wir nun in dieser Ausgabe abschließend die Vernetzung und Applikationen der Mikrocomputer sowie einige periphere Geräte betrachten.

Computernetze

Das Kombinat Robotron stellte sein lokales Netz **ROLANET 1**, das bereits auf der Leipziger Frühjahrsmesse 1987 zu sehen war, aus. Diesmal wurde eine Kopplung mit der digitalen Nebenstellenzentrale NZ 400 D (Kombinat Nachrichtenelektronik) über einen PC 1715 gezeigt. Dieser PC 1715 war über zwei getrennte serielle Schnittstellen sowohl mit dem ROLANET 1 als auch mit der NZ 400 D derart gekoppelt, daß Rechner in der NZ 400 D mit Rechnern im ROLANET 1 direkt kommunizieren konnten.

Zur Realisierung eines ROLANET 1 (Bruttoübertragungsrate: 500 KBaud) werden die Lokalnetzcontroller LNC 1, die als Erweiterungskarten in den Rechnern stecken, und die Transceiver vom Typ TCR K 8601 (Bild 1), die das Koppelglied zwischen Koaxialkabel und Stichleitung zum LNC 1 darstellen, benötigt. Zusätzlich zu den Lokalnetzcontrollern für die Rechner PC 1715, A 7100 (wird softwaremäßig nicht unterstützt), K 1630 sowie A 5120 sind jetzt auch Controller für den 32-Bit-Minirechner K 1840 sowie die 16-Bit-Mikrorechner EC 1834 und A 7150 verfügbar.

Die digitale Nebenstellenzentrale NZ 400 D ermöglicht neben dem Anschluß von Telefonen u. a. auch die Kopplung von Computern. Die Computerschnittstelle wird über einen zusätzlichen digitalen Teilnehmersatz in der NZ 400 D sowie über die **Datenanschlußeinrichtung DAE** (Bild 2) realisiert. Maximal 15 % der Teilnehmeranschlußzahl können zusätzlich für Computer bereitgestellt werden.

Die NZ 400 D wird in 4 Typen mit 64, 128, 256 oder 384 Anschlußeinheiten gefertigt. In Leipzig war die NZ 400 D/128 (Bild 3) mit 64...112 Teilnehmern und maximal 16 Hauptanschlußleitungen zu sehen. Die Schnittstelle X.21 bzw. X.21 bis zur Datenanschlußeinrichtung ermöglicht eine Datenübertragungsgeschwindigkeit von 2,4; 4,8 oder 9,6 KBaud und eine Länge der Anschlußleitung (2-Draht-Kupfer-Leitung) von maximal 3 km.

Auch das SCOM-LAN der Ingenieurhochschule für Seefahrt Warnemünde/Wustrow war bereits 1987 auf der Frühjahrsmesse zu sehen. Neu angeboten wurden hierfür die Baugruppen **SCOM-PROLAN 1834**, eine LAN-Slotkarte für den EC 1834, sowie die **SCOM-LANBOX**. Die LANBOX (Bild 4) beinhaltet die PROLAN-Slotkarte sowie ein separates Netzteil und ist somit für alle Rechner mit synchroner V.24-Schnittstelle einsetzbar. Mit dem optischen Empfänger OE 500 und dem optischen Sender OS 500 vom Kombinat EAW ermöglicht die LANBOX auch den Übergang zwischen Koaxialkabel und Lichtwellenleiter. PROLAN und LANBOX arbeiten mit einer maximalen Bruttoübertragungsrate von 153,6 KBaud. Sie beinhalten weiterhin eine V.24-Schnittstelle und optional einen Feldbusanschluß (TU Magdeburg; siehe auch: Intelligente dezentrale Prozeßkoppelmodule, Feingerätetechnik 8/1987, S. 360).

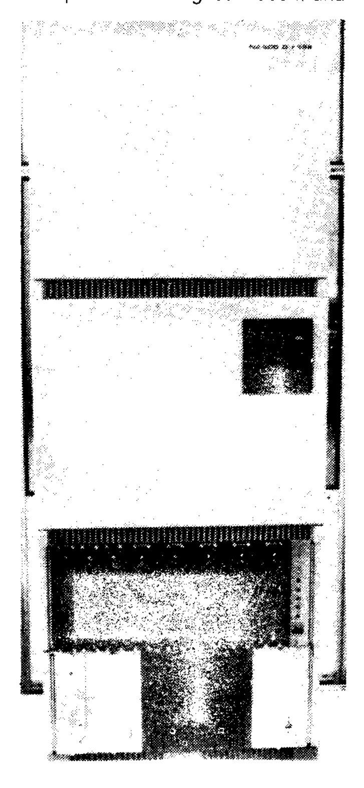
Auch zahlreiche ausländische Aussteller boten Vernetzungsmöglichkeiten ihrer Computer an. Das bulgarische Außenhandelsunternehmen ISOTIMPEX beispielsweise für ihre Personalcomputer das Microring-LAN und **Micro STAR** (Farbbild 1; alle Farbbilder siehe 3. Umschlagseite). Letzteres besitzt – wie der Name ausdrückt – eine Stern-Topologie. An die Zentralstation können über eine verdrillte Leitung bis zu 8 Stationen angeschlossen werden. Die Übertragungsgeschwindigkeit beträgt bei Entfernungen bis zu 15 m 115 200 Bit/s und bis zu 300 m 19 600 Bit/s.

Die ungarische Videotonwerke zeigten, wie an ihren Minirechner EC 1011 über einen Konzentrator (Z-80-gesteuerter Multiplexer mit 8 seriellen und einer parallelen Schnittstelle, 64 KByte RAM, 12 KByte EPROM) bis zu 4 Personalcomputer angeschlossen werden können (Bild 5). In diesem **TTRNET** haben unterschiedliche 8- oder 16-Bit-PCs (z. B. PC 1715, VT 160, EC 1834) die Möglichkeit des Zugriffs auf die TTR-Datenbank im EC 1011. Gesteuert wird das Ganze vom Datenbanksystem TTR 654.

PC-Applikationen

Während Personalcomputer am Anfang ihrer Entwicklung vor allem im ökonomischen Bereich eingesetzt wurden, erlaubt ihre heutige Leistungsfähigkeit, sie auch mit Aufgaben des Entwurfes, der Steuerung oder der Bildverarbeitung zu betrauen.

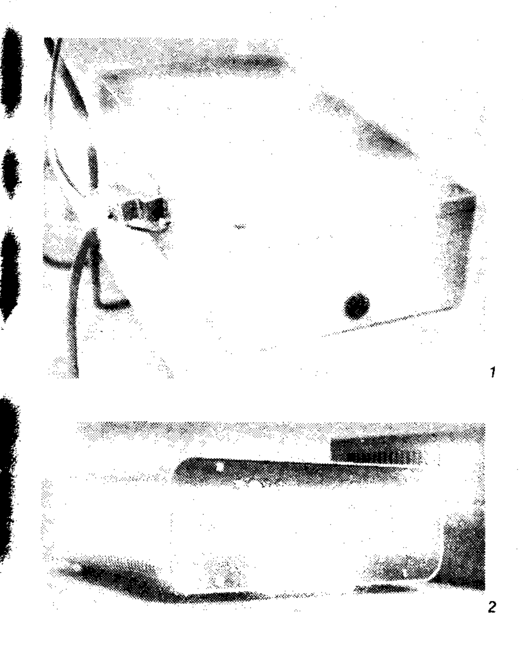
Das Kombinat Robotron entwickelte aus den Erfahrungen mit dem Bildverarbeitungssystem A 6471 für den EC 1834 ein automatisiertes, digitales System **IMAGE-C**. Die Hardwarebasis umfaßt den PC mit 640 KByte RAM, einer Floppy mit 720 KByte und einer Festplatte von 20 MByte; darüber hinaus die Bildverarbeitungseinheit K 7067.15 (Bildspeicher 768×512×8 Bit). Die Bildeingabe erfolgt über den Videoeingang der K 7067, die Ausgabe über Farbmonitor und Grafikdrucker. Als Vorteil wird hervorgehoben, daß dem Anwender eine Benutzeroberfläche mit Menüsystem ohne Notwendigkeit eigener Programmierung zur Verfügung steht. Das Bildverarbeitungssoftwarepaket ist in C geschrieben, und



3

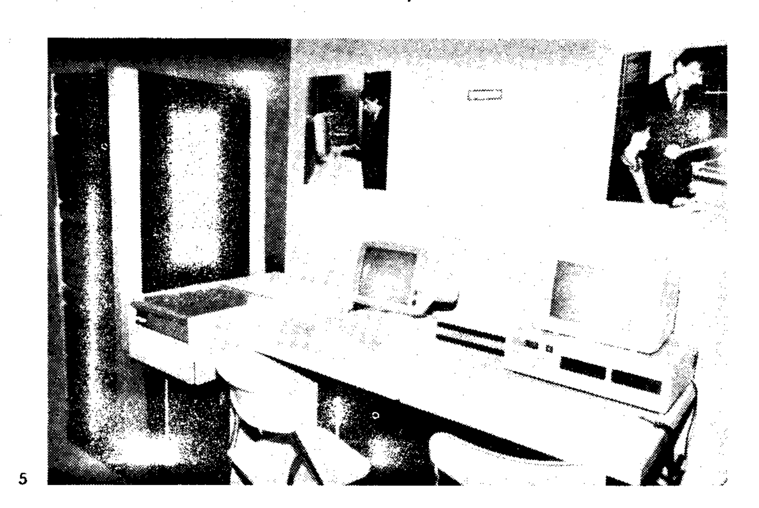


4

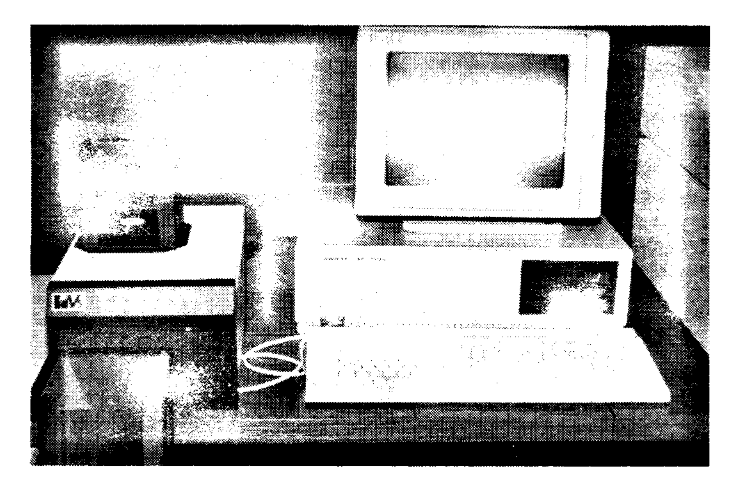


1

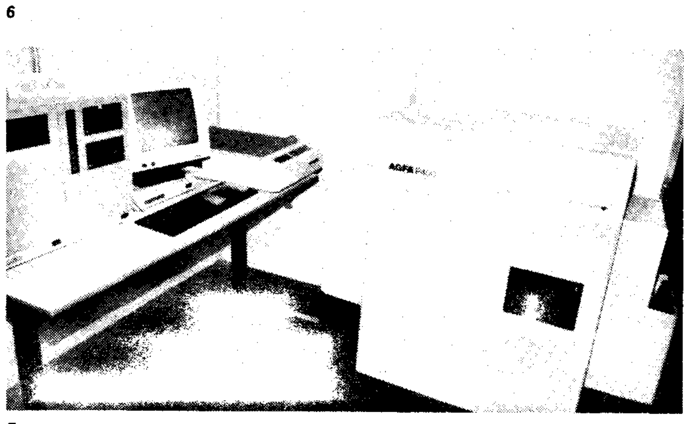
2



5



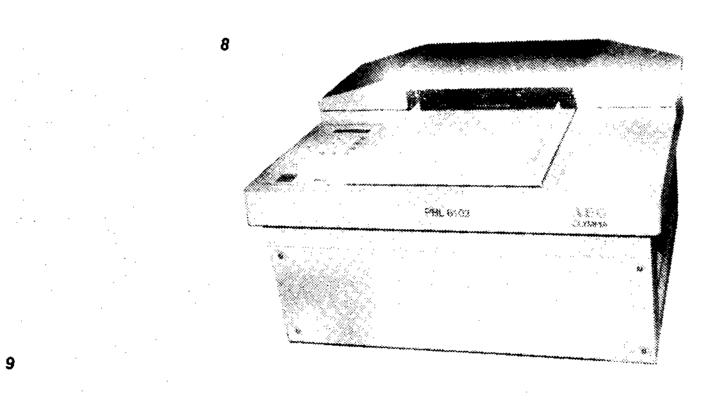
6



7

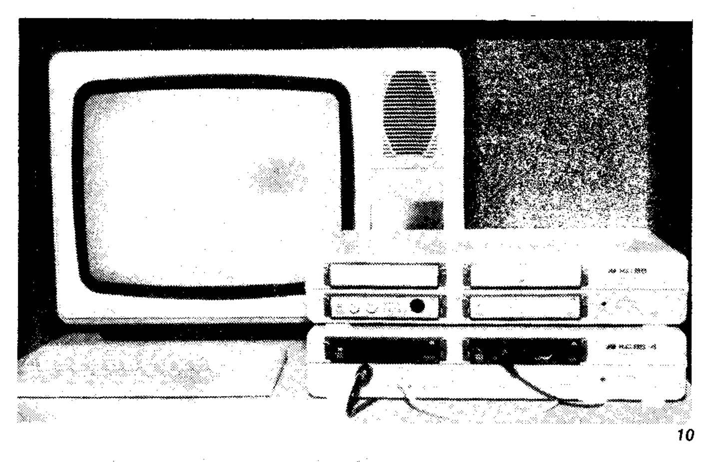
die Standardbibliotheken dieses Programmiersystems wurden in das System eingebunden.
Verschiedene Angebote enthielten den PC als Bestandteil einer kompletten Anwendungslösung mit spezieller Peripherie. Dazu nachfolgend einige Beispiele.
Als vorgeschalteter Satzrechner fungierte bei Robotron der EC 1834, verbunden mit dem **Laserbelichter CLS 2090 Laserjet** der Computer Gesellschaft Konstanz mbH (Farbbild 2). Der Ganzseitenbelichter übernimmt vom Satzrechner den Inhalt einer Seite und bereitet ihn so auf, daß die Seite in einem Zug von dem HeNe-Glaslaser belichtet werden kann. Basis sind dafür ein Multi-Mikroprozessorsystem mit 8085, 8086 und Arithmetikprozessor 8087 sowie ein interner Speicher von 1 MByte. Für das Programm, die Zwischenspeicherung und die digitalisierten Fonts (Zeichensätze) sind zwei bis vier 5¹/₄-Zoll-Diskettenlaufwerke mit je 1,2 MByte vorgesehen. Da pro Fontdiskette 20 bis 30 Masterfonts gespeichert werden können, stehen bis zu 90 Schriften in direktem Zugriff. Der Belichter arbeitet mit einer Geschwindigkeit von etwa 87 mm/Minute (etwa 32 000 Zeichen/h) und einer Auflösung von horizontal 1446 Linien/Zoll und vertikal 723 Linien/Zoll bei Pica-Schrift. Entwickelt wurde er speziell für die vielseitigen Bedürfnisse kleiner und mittlerer Betriebe.
Ebenfalls als Host-Rechner arbeitete der Robotron EC 1834 in Verbindung mit einer Disketten-Duplizierstation der Inverdata Electronics GmbH (Bild 6). Damit sind das automatische Formatieren, Duplizieren sowie Stapellesen und -schreiben möglich. Die Steuerung der Loaderfunktion erfolgt über einen sehr einfachen Befehlssatz; die Daten werden über die Standardlaufwerkschnittstelle transferiert. Ein Separier- bzw. Rollenmechanismus transportiert die Diskette vom Zufuhrmagazin in das Laufwerk der Station und nach der Bearbeitung in das jeweils angesprochene Ablagefach. Die abgebildete Station **I.E.-V5** kann im Zufuhrfach 30 oder 100 Disketten, in der Ablage „gut" 30 oder 100 Disketten und in der Ablage „fehlerhaft" 30 Disketten aufnehmen. Das Format beträgt 5¹/₄ Zoll mit 48 oder 96 tpi, HD.
Daß selbstverständlich auch nichtkompatible Mikrocomputer Bestandteile von PC-Anwendungslösungen sein können, zeigte die Centroimpex Industrievertretung anhand der Kopplung des P 8000 mit dem Agfa-Scanner S 200 PC und dem neuartigen elektronischen Drucker Agfa P 400 (Bild 7) in einem Textsystem. Der **S 200 PC** ist ein Flachbett-Scanner mit einer A4-Arbeitsfläche (215×350 mm²), das heißt, die Vorlage wird wie bei einem Kopierer mit der Bildseite nach unten flach auf eine Glasplatte gelegt. Dicke, Größe und Art der Vorlage beeinflussen die Erfassungsqualität nicht. Fast 2000 Sensoren tasten die Vorlage zeilenweise ab, wobei jeder der Sensoren 64 verschiedene Graustufen unterscheiden kann. Dabei bewegt sich der Abtastkopf gleichmäßig und ohne Unterbrechung über das Original. Es ergeben sich dann pro Bildseite je nach Vorlage 4 bis 16 Millionen Bildpunkte, die in ein spezielles Datenformat umgewandelt und bei Bedarf auch komprimiert werden können. Dennoch fallen – bei der hohen Arbeitsgeschwindigkeit von 3 Sekunden pro A4-Seite – so viele Daten an, daß im Scanner ein Zwischenspeicher von 128 KByte (etwa 1 Million Bildpunkte) vorgesehen wurde. Von dort werden die Daten über eine serielle asynchrone RS-232-Schnittstelle mit bis zu 9600 Baud oder über eine parallele Schnittstelle mit bis zu 200 KByte je Sekunde in den Computer übertragen.
Für die Ausgabe zum Beispiel der über den Scanner erfaßten und im Computer bearbeiteten Informationen liefert Agfa als erster Hersteller einen elektrofotografischen Drucker, bei dem die Zeichen über eine Leuchtdiodenreihe ohne Einbeziehung beweglicher Teile auf Normalpapier oder Folie übertragen werden. Die LED-Zeile erlaubt eine Auflösung von jeweils 16 Punkten/mm in der Waagerechten und Senkrechten. Wie beim Laserdrucker sind hierbei keine Durchschläge möglich, jeder Ausdruck ist praktisch ein Original. Die Stärke des **P 400** ist zwar das Drucken von Texten, ebenso problemlos können jedoch auch Grafiken, Schraffuren und Symbole ausgegeben werden. Zudem lassen sich über den Scanner erfaßte Fotos, Zeichnungen und Bildelemente in den elektronischen Druck einfügen. Ein 2-MByte-EPROM und 4-MByte-RAM dienen als Zeichenspeicher zur Aufnahme von Schriftsätzen; zusätzlich stehen zwei im Drucker integrierte 5¹/₄-Zoll-Diskettenlaufwerke und ein Winchester-Festplattenlaufwerk zur Verfügung. Somit kann man beispielsweise ganze Formulare als Maske abspeichern. Die Druckgeschwindigkeit des P 400 liegt bei etwa 18 A4-Seiten je Minute.
Auch die Handelsgesellschaft Transcommerz (die man zu früheren Messen übrigens schon mit einer größeren Computerpräsentation sah) zeigte eine Möglichkeit der modernen Textverarbeitung. In Verbindung mit einem IBM-PC wurde demonstriert, wie der AEG-Blattleser **PBL 6103** (Bild 8) als Eingabegerät für Textsysteme genutzt werden kann. Das Charakteristische an dem Blattleser ist, daß er maschinengeschriebene Vorlagen (mit den 6 gängigsten Schreibmaschinen-Schriften) automatisch in ein Textsystem, Datennetz oder eine Fotosatzanlage übertragen kann. Dazu wird die Vorlage mittels 2048 Fotodioden in einer Zeile Zeichen für Zeichen gelesen, und die Daten werden über eine V.24/V.28-Schnittstelle mit 110 bis 19 200 Bit/s übertragen. (Im Gegensatz zum Agfa-Scanner bewegt sich beim PBL 6103 die Transportvorrichtung mit dem Datenträger an der Fotodiodenzeile vorbei.) Mindestens für eine A4-Seite können die Daten im Pufferspeicher abgelegt werden. Die Verarbeitungsleistung liegt bei über 150 Zeichen/s so daß sich je nach Textumfang und Druckqualität ein typischer Durchsatz von 25 bis 40 Sekunden je Blatt ergibt.
Im ersten Teil unseres Messeberichtes hatten wir bereits auf die CAD-Station des polnischen Werkes ZIPO hingewiesen. ZIPO ist ein in Gdansk ansässiger „Betrieb für Informatik in der Schiffbauindustrie", der sich auf Dienstleistungen der Informationsverarbeitung spezialisiert hat. In Leipzig präsentierte er das Leiterplattenentwurfssystem **PCAD** auf der Hardwarebasis eines PC XT/AT-kompatiblen Rechners mit dem Prozessor 80386 (mit 20 MHz), 640 KByte RAM, Hercules-Monochrom-Monitor und Farbgrafik-Monitor (Farbbild 3).
Letzteres können sowohl EGA- als auch 1024×1014- oder 1280×1024-Monitore sein. Die Festplattenkapazität beträgt mindestens 20 MByte. Die Software umfaßt eine Bauteilbibliothek zur Erstellung von Schaltplänen und Leiterplatten mit etwa 400 TTL- und 100 diskreten Elementen, etwa 160 CMOS-Schaltungen und etwa 150 hochintegrierten Schaltungen der Intel-Reihe. Mehr als 15 Programme unterstützen den automatischen Entwurf, die Plazierung und Netzwerkanalyse bis zur Steuerung der Fotoplotter und Ausgabe der Dokumentation. Hard- und Software sollen sowohl einzeln als auch geschlossen zu beziehen sein.
Einen **CAD-Arbeitsplatz** stellte ebenfalls Rank Xerox Ltd. vor, eine Firma, die seit mehr als 20 Jahren in der DDR tätig ist. Basis des Arbeitsplatzes war der Rank Xerox **OPat PLUS** mit 80287-Koprozessor und der neuesten Version von AutoCAD,

8



9





10

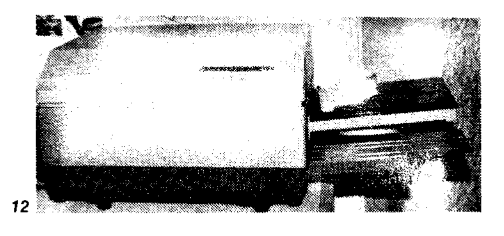


11

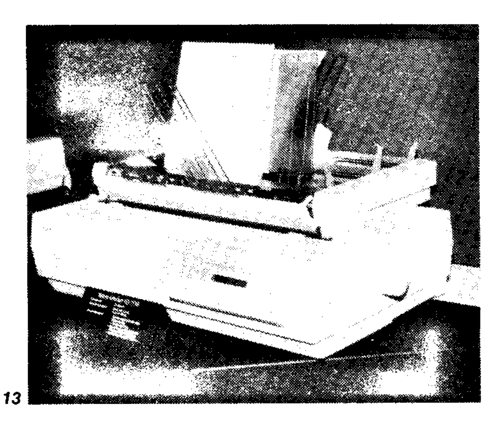
der Ausgabe 9.0 als Nachfolger der Version 2.6 (Farbbild 4). Die Bedienerführung wurde unter anderem durch Dialogfenster verbessert, und eine vollständige Portabilität der Dateien ist insbesondere in Netzwerkanwendungen nützlich. Zu der Konfiguration gehörten ein EGA-Bildschirm, ein hochauflösender 20-Zoll-Farbgrafik-Bildschirm (1024×768 Pixel), zur Eingabe ein Grafiktablett (280×430 mm²) mit Lupe und 4 Tasten sowie ein 8-Farben-Stiftplotter DMP 29 von Houston Instruments.
Die gute Position unter den Desktop-Publishing-Anbietern unterstrich Rank Xerox auch mit einer **DTP-Lösung**, die den neuen Laserdrucker **4046** einbezog (Bild 9). Der 4046, der wie sein Vorgänger 4045 auch als Kopierer verwendbar ist, wurde diesem gegenüber mit folgenden neuen Funktionsmöglichkeiten ausgestattet: automatische Papierumschaltung, Non-stop-Druck und adressierbare Papierbehälter.
Zur Eingabe der Daten wurde ein Microtek-MSF-300C-Scanner genutzt, der bei einer Auflösung von 300×300 dpi (11,8 Punkte/mm) die Daten der Vorlage mit 9600, 19200 oder 57600 Baud über eine RS-232C/RS-422-Schnittstelle dem Computer übermittelt. Als Software für dieses DTP-System wird (selbstverständlich) der Rank Xerox Ventura Publisher verwendet.
Weitere PC-Lösungen zeigte Rank Xerox mit der Kopplung des Robotron A 7150 – RX 4045 als DTP-System (siehe auch MP 3/88, 2. Umschlagseite) und mit einem lokalen Netz.
Eine nützliche Erweiterung, um die 640-KByte-Grenze bei PCs unter dem Betriebssystem MS-DOS überschreiten zu können, wurde von Frisbie Data Systems mit der **RYBS-AMS-HIcard** vorgestellt (Farbbild 5). Während es von Lotus, Intel und Microsoft für den gleichen Zweck als Softwarelösung den sogenannten LIM-Standard gibt, ist die HIcard eine kurze 256-KByte-RAM-Steckkarte, die den Speicherbereich eines IBM-PC XT/AT oder Kompatiblen bis zu 896 KByte erweitert. Über eine einfache menügetriebene Software können RAM-Disks, Druckerspooler, Terminate-Stay-Resident-(TSR-) Programme, Netzwerktreibersoftware, Communication Programme und E-Mail installiert werden. Die HIcard ist lauffähig mit allen PC/MS-DOS Versionen ab 2.00 und auch den neuesten Versionen integrierter Software wie Symphony, Lotus oder dBASE III.

Kleincomputer und Peripherie

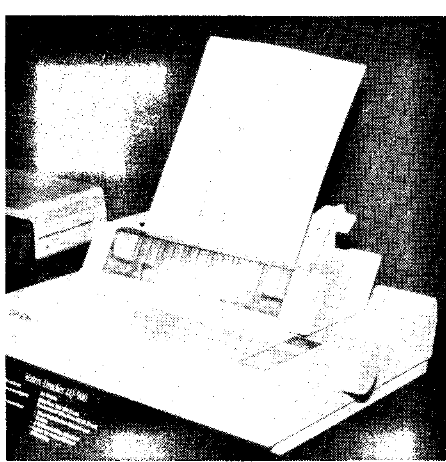
Das Kombinat Mikroelektronik stellte den Kleincomputer **KC 85/4** (Bild 10, hier mit Bustreiberaufsatz D002 und Farbmonitor **KCM 38-1**), eine Weiterentwicklung des KC 85/3, in verändertem Design vor. Er besitzt gegenüber seinem Vorgänger einen erweiterten RAM von 128 KByte (davon 64 KByte Bildwiederholspeicher IRM), ein erweitertes Betriebssystem sowie eine erhöhte Farbauflösung (1 mal 8 Bit). Zwischen zwei Bildebenen kann umgeschaltet werden. Bildschirmzugriffe durch den Rechner werden vom Betrachter nicht mehr wahrgenommen. Der HF-Modulator für das Betreiben eines Fernsehgerätes über seinen Antenneneingang arbeitet auf dem UHF-Kanal 36. Mit 32 KByte des IRM sowie durch Stecken von einem 16-KByte-RAM-Modul (im Gerät) und zwei 64-KByte-RAM-Moduln (Modulschächte) kann eine RAM-Disk von 176 KByte installiert werden.
Für den Kleincomputer KC 85/4 (/2,/3) des Kombinates Mikroelektronik wurden der Erweiterungsaufsatz **D004** (FLOPPY DISK BASIS) für die Floppy-Disk-Ansteuerung sowie das Beistellgerät **FLOPPY DISK DRIVE** (5¼-Zoll-Laufwerk K 5601 bzw. 1.6) vorgestellt (Bild 11, hier ebenfalls mit 42-cm-Farbmonitor KCM 38-1). Der Aufsatz beinhaltet eine CPU UA 880 D, 64 KByte RAM und einen Floppy-Disk-Controller U 8272 D. Er ermöglicht weiterhin die Installierung einer RAM-Disk von 288 KByte durch Stecken von zwei 16-KByte-RAM-Moduln (im Grundgerät) und vier 64-KByte-RAM-Moduln (Modulschächte) sowie 40 bzw. 80 Zeichen pro Bildschirmzeile (umschaltbar).
Der Modul M036 **128 KBYTE SEGMENTED RAM** mit einem Hybridschaltkreis vom Typ 4734 wurde vom Kombinat Keramische Werke Hermsdorf für den KC 85/4 (/2,/3) angeboten. Diese Baugruppe ermöglicht dem KC 85 den Zugriff auf 8 Segmente mit einer Größe von je 16 KByte.
Von den peripheren Geräten wollen wir zunächst einige interessante Druckerneuigkeiten vorstellen.
Das Kombinat Robotron offerierte neben seinen bekannten Serien-, Nadel- und Thermodruckern erstmals einen Laserdrucker. Der **EC 7230** (Farbbild 6) ist für ESER-Anlagen vorgesehen und liefert bis zu 20 A4-Seiten (Normalpapier) in der Minute. Die Auflösung beträgt 240×240 Punkte/Zoll. Die Druckausgabe erfolgt seitenweise auf Einzelblätter. Da bei einem Laserdrucker keine Durchschläge möglich sind, können geräteintern durch wiederholte Ausgabe bis zu 8 Exemplare einer Seite erstellt werden. Der EC 7230 wird in verschiedenen Ausstattungsvarianten geliefert, denn nicht in jedem Fall wird beim Anwender das volle Leistungsspektrum benötigt.



12

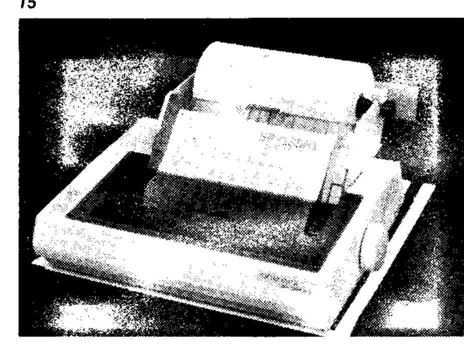


13



14

15



16



Was bei Laserdruckern noch Zukunft ist, demonstrierte Tektronix mit dem Grafikdrucker **TEK 4693 D** (Bild 12): die Möglichkeit, im 4-Farbdruck über 16 Millionen Farben in einer Auflösung von 300 Punkten/Zoll darstellen zu können. Basis ist die Kombination einer Thermotransfer-Drucktechnologie mit speziellen Bildaufbereitungsalgorithmen. Sie erlauben unter anderem schattierte Festkörpermodellierungen, grafische Bildverarbeitung, Präsentationsgrafiken, medizinische Grafiken oder Leiterplattenentwicklung. Die hohe Flexibilität wird durch einen internen Motorola-68020-Mikroprozessor (32 Bit) erreicht. Ein 4-MByte-Bildspeicher und die spezielle Tektronik-Hochgeschwindigkeits-Parallelschnittstelle ermöglichen es, Bilder vom PC mit 800 KByte/s zu übertragen. Nach etwa 20 Sekunden für Farb- und Dimensionierungsmanipulationen im Drucker erfolgt der Ausdruck dann innerhalb 60 bis 100 Sekunden. (Das Farbbild 8 zeigt einen Demonstrationsausdruck.) In der 12-MByte-Konfiguration kann der 4693 D drei Bilder gleichzeitig manipulieren. Während ein Bild ausgedruckt wird, verarbeitet der Drucker ein zweites und hält ein drittes in der Warteschlange.

Epson zeigte mit dem 24-Nadeldrukker **LQ-2550** (Bild 13), bei dem der Farbdruck nunmehr serienmäßig ist, daß auch die Anschlagdrucker ein beachtliches Leistungsniveau erreichen. Die Grafikauflösung liegt mit 360×360 Punkten/Zoll über der vieler Laserdrucker, und die Geschwindigkeit konnte dank des beschleunigten Zeilenvorschubs bei Schnelldruck (Draft) auf 400 und bei Schönschrift (LQ) auf 133 Zeichen/Sekunde gesteigert werden. Durch das Breitformat ist der LQ-2550 auch für CAD-Ausdrucke in A3 geeignet. Daß der Drucker alle Papierfunktionen vom Einzug bis zum wechselseitigen Zustellen von Einzelblatt und Endlospapier vollautomatisch durchführt, eine Abreißautomatik besitzt, sich selbsttätig auf die Papierstärke einstellt und zahlreiche ladbare Zeichensätze bietet, ist bei einem Epson-Drucker dieser Leistungsklasse fast selbstverständlich.

Mit dem **LQ-500** und dem Slogan „Der 24-Nadel-Drucker fürs kleine Budget" wollte der gleiche Hersteller demonstrieren, daß er auch am unteren Ende der Angebotsskala Qualität zu bieten hat. Als Käuferschicht werden den Besitzer von Home-Computern und sogenannten Low-Cost-PCs angesehen. Der LQ-500 (Bild 14) erzielt aufgrund der 24 Nadeln im Druckkopf nicht nur Fast-Brief-Qualität (NLQ), sondern echte Schönschrift (LQ). Die Druckgeschwindigkeit beträgt bei dieser Betriebsart 60 Zeichen/Sekunde; bei Schnelldruck sind es 180 Zeichen/Sekunde. In der Grundausstattung enthält der LQ-500 die Schnellschrift New Draft und die beiden Schönschriftarten Roman und Sans Serif. Weitere Schriftarten sind als Steckmoduln erhältlich. Als Standard wird der Drucker mit halbautomatischem Einzelblatteinzug, Friktionswalze und Aufsatzzugtraktor geliefert, als Option mit automatischem Einzelblattmagazin.

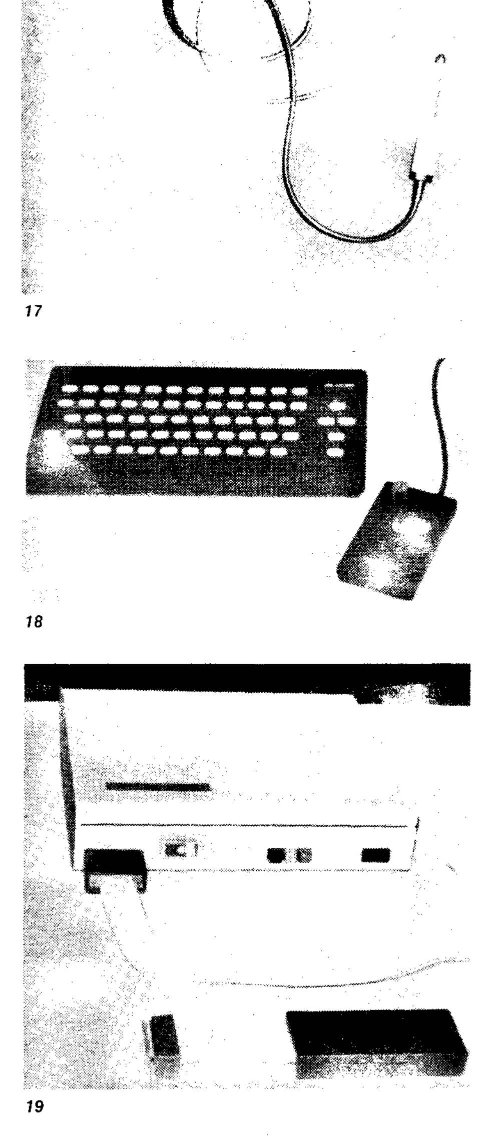
Erstmals in Leipzig vertreten war der namhafte Druckerproduzent Seikosha (der den zur Zeit vielleicht schnellsten Nadel-Matrixdrucker der Welt herstellt: den SB 10 mit 800 Zeichen/s) mit einer Auswahl seiner Erzeugnisse. Wir wollen hier den **SP-185 AI** vorstellen (Bild 15), der vom Hersteller ebenfalls als Small-Business- bzw. Family-Printer eingestuft wird. Das Besondere am SP-185-AI ist, daß er papiersparend arbeitet, indem beim Blattwechsel kein automatischer Papiervorschub vollzogen wird. Er hat darüber hinaus einen Rollenpapierhalter, in welchen sich problemlos beispielsweise Telexpapierrollen einlegen lassen. Weitere Daten sind: 9-Nadel-Drucker, 100 Zeichen/s bei Schnellschrift, 20 Zeichen/s bei NLQ, standardmäßig Parallel-Interface, kompatibel zum Epson-FX-80- und IBM-5152-Standard, mehrere Schriftarten sowie 128 ladbare Zeichensätze.

Als ebenfalls mögliches Ausgabegerät für Heimcomputer wurde vom Kombinat Robotron die elektronische Kleinschreibmaschine **Erika 3004 electronic** präsentiert (Bild 16), die mit Typenrad natürlich Schönschrift liefert. Die Schreibgeschwindigkeit liegt bei 10 Zeichen/s, es gibt Tasten für zahlreiche Zusatzfunktionen und einen 20 Zeichen umfassenden Korrekturspeicher. Bemerkenswert an der 3004 ist, daß über die Systemschnittstelle der Maschine der Anschluß einer Interfacebox IF 6000 (im Bild rechts) für Commodore/Centronics oder für V.24 zur Kopplung an einen Home- oder Personalcomputer gegeben ist. Das gleiche gilt auch für das Modell Erika 3005 electronic, welches unter anderem mit einem größeren Korrekturspeicher und einem 7,5-KByte-Textspeicher jedoch komfortabler ist.

Nach den Druckern zum Schluß noch zu einigen weiteren E/A-Geräten.

Tektronix liefert für wissenschaftliche und technische Anwendungen unter anderem ein leistungsstarkes Farbterminal mit VT-100-kompatibler Alphanumerik. Das **GS 4209** (Farbbild 7) hat einen Standardspeicher von 512 KByte und eine eigene Intelligenz. Es erzeugt Grafiken in 16 gleichzeitig darstellbaren Farben bei einer Auflösung von 640×480 Bildpunkten und 60 Hz Bildwiederholrate. Es gibt Zoom- und Panfunktion; per Taste können beliebige Bildausschnitte vergrößert und mit 1024 ×1024 adressierbaren Punkten dargestellt werden. Darüber hinaus wird das Zeichnen auf unabhängigen Ebenen unterstützt.

Das ungarische Unternehmen für Elektronikentwicklung ERFI zeigte ein mobiles Datenerfassungsterminal im Taschenformat, das **LD-CF** (Bild 17). Es ist mikroprozessorgesteuert und kann 30 bis 112 KByte Daten im RAM speichern. Im 2 bis 8 KByte großen EPROM ist das Anwenderprogramm abgelegt. Über erforderliche Handlungen und den Zustand des Gerätes informiert die 16stellige LC-Anzeige. Die Daten werden entweder über die alphanumerische Tastatur oder über einen Lesestift eingegeben. Sie können dann bei Bedarf über eine V.24-Schnittstelle mit 300 bis 9600 Baud mittels Netzteil/Bit-Rate-Generator LD-TB in den Computer übertragen werden.



17

18

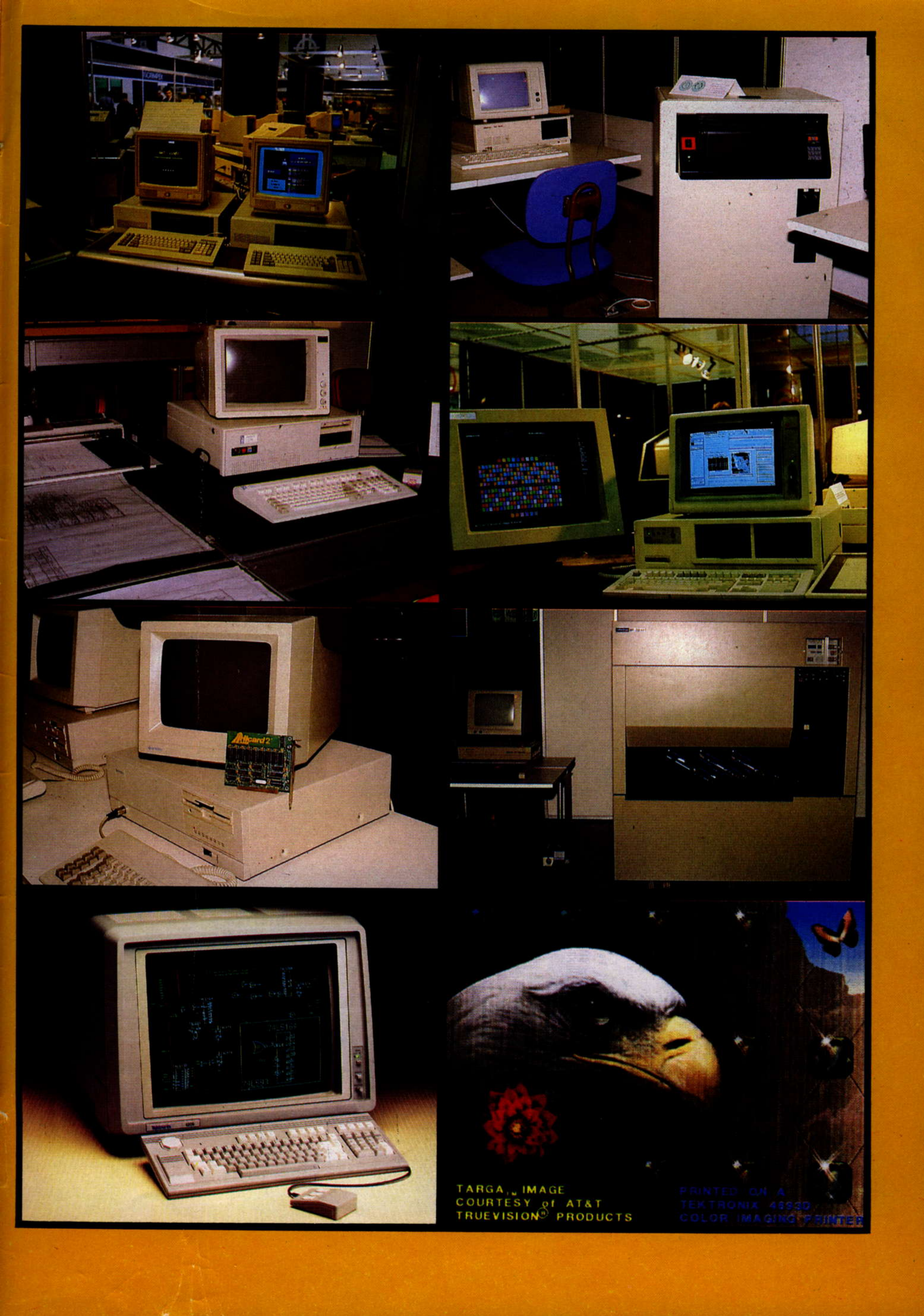
19

Vom Kombinat VEB Elektro-Apparate-Werke Berlin-Treptow wurde ein kontaktloser **Joystick** für verschiedene Mikrocomputer (z.B. KC-85-Reihe, Z-1013, P 8000) gezeigt (Bild 18). Der analoge Joystick besitzt keine Potentiometer und kaum Mechanik, sondern statt dessen magnetoresistive Sensoren. Diese zur analogen Messung schwacher Magnetfelder geeigneten Bauelemente werden zur kontaktlosen Strom-, Positions- und Winkelmessung genutzt. Der Joystick hat eine Auflösung von über 100×100 Schritten und ist für alle Computer mit 8-Bit-E/A-Port geeignet.

Ebenfalls vom KEAW war der Einchiprechner-Emulator 8000 zu sehen (Bild 19). Er stellt ein Zubehör zum Programmier- und Entwicklungssystem P 8000 dar (s. MP 3/87), kann aber auch mit anderen Rechnern über zwei V.24-Schnittstellen unter den Betriebssystemen UDOS und UNIX betrieben werden. Der Emulator ist eine Weiterentwicklung des U-881-/U-882-Emulators vom ZFT des KEAW. Er ist für die Hard- und Softwaretestung von Einchiprechnersteuerungen mit den EMR U 881, U 882, U 8611, U 884 sowie U 886 geeignet. Es können Programme mit einer Länge von 4 KByte (U 886: 2 KByte) getestet werden. Die Software kann von Diskette in einen 4-KByte-RAM geladen oder mit einem EPROM (U 2716 oder U 2732) auf die Schwenkhebelfassung an der Gerätevorderseite gesteckt werden. Zur Verbindung des Emulators mit den Anwendersystemen werden für die verschiedenen Rechnertypen 4 Tastkopf- und 3 Adapterversionen geliefert. Es ist Echtzeittestung mit Breakpoints, Schrittbetrieb sowie Tracebetrieb möglich.

Hans Weiß/Herbert Hemke

Fotos: Weiß (13), Hemke (5), Werkfoto



TARGA™ IMAGE
COURTESY of AT&T
TRUEVISION® PRODUCTS
PRINTED ON A
TEKTRONIX 4693D
COLOR IMAGING PRINTER

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 8
S. 225–256



Technik international



Der PAC 286 (Personal Advanced Computer) mit 80286-Prozessor, 1 bis 16 MByte Hauptspeicher und neuartigem Memory Management System (MMS) ist als Tower ausgeführt, der bis zu zwei der transportablen Festplatten-Module aufnehmen kann.

Werkfoto

Die mobile Festplatte

Für den Fachmann mag die Überschrift einen Widerspruch in sich bergen. Schließlich ist es für Festplatten charakteristisch, daß sie im Laufwerk und dieses wiederum im Computer fest installiert, also nicht mobil sind. Der Vorteil ist offensichtlich, wenn man an die höheren Kapazitäten von Festplattenspeichern gegenüber denen von mobilen Platten – beispielsweise Disketten – denkt. Festplattenlaufwerke können nämlich aufgrund ihrer Konstruktion mit einer viel höheren Präzision der Paarung Schreib-/Lesekopf – Platte hergestellt werden als Diskettenlaufwerke. Allerdings haben sie den unübersehbaren Nachteil der aufwendigen Datensicherung und Archivierung.

Ein neues Konzept

Einer der „Väter" des Personal-Computers und heutige Präsident der Firma Tandon verwirklichte nunmehr die Idee, die Festplatte als Datenträger mitsamt dem Laufwerk als Modul transportabel zu gestalten. Somit kann der Vorteil der Disketten – die Mobilität – mit dem der Festplatten – hohe Speicherkapazität – verbunden werden. Als Weltpremiere wurde zur CeBIT 1987 dieses neue Wechsel-Festplattensystem als Bestandteil des PC AT-kompatiblen PAC 286 vorgestellt. Es ist ein 1,2 kg schweres, 18 × 12 × 6 cm³ großes Gehäuse, das ähnlich wie eine Videokassette in einen der beiden Modulschächte des PAC eingelegt wird. Der Computer übernimmt dann den automatischen Einzug und das korrekte Ausrichten. Der „Auswurf" erfolgt mittels Tastendruck. Auf diese Art wird es möglich, statt Disketten Personal Data Pacs (PDP) – so werden die Module genannt – als persönliche Datenträger zu verwenden, die jedoch auch umfangreiche Programme geschlossen aufnehmen. Oder große Datenbestände werden unmittelbar nach Beenden der Arbeit entnommen und sicher verwahrt. Da der PAC 286 zwei Einschübe besitzt, können die Daten auch in etwa 100 Sekunden von einem PDP auf das andere als Back-up übertragen werden.

Die Technologie

Kernstück des PDP ist ein 3½-Zoll-Winchester-Plattenlaufwerk mit 32 MByte (formatiert), das für den Einsatz unter extremen Bedingungen ausgelegt wurde. Bei traditionellen Festplatten wird beim Ausschalten der Schreib-/Lesekopf zurückgezogen und in einer nicht für die Datenspeicherung vorgesehenen Zone „geparkt". Da die Köpfe sehr empfindlich sind, können Erschütterungen zu Schäden am Mechanismus und sogar zum Datenverlust führen. Beim PDP dagegen werden die Köpfe mit dem Auswerfen des Moduls automatisch von der Platte weggezogen und fest verankert. Außerdem ist das Laufwerk in einem starren Stahlrahmen mit vier schnell reagierenden Stoßdämpfern aufgehängt, und dieser Korpus wird zusätzlich von einem robusten, stoßunempfindlichen Plastmantel umgeben. Durch diese Maßnahmen sollen – wie Testberichte bestätigen – mehrmaliges Fallenlassen bis hin zum mutwilligen Beschädigen des Gehäuses keinen Einfluß auf die Funktionsfähigkeit der Festplatte haben. Zur hohen Funktionssicherheit trägt auch bei, daß sich im PDP selbst keine anfällige Steuerlogik befindet.
Weiterhin werden als leistungssteigernde Maßnahmen genannt: spezielle Controller-Karte, die die neue RLL-Technologie (run lenght limited) nutzt – diese erlaubt eine im Vergleich zum MFM-Speicherverfahren bis zu 50 Prozent höhere Kapazität des Datenträgers; 128 KByte RAM sowie Cache-Algorithmen im ROM zur Reduzierung der Zugriffszeiten; die Plattenverzeichnisse sind in der Mitte der einzelnen Platten abgelegt, so daß sich die durchschnittliche Daten-Suchzeit zusätzlich verringert, sie beträgt weniger als 40 ms.

Auch an Kompatible gedacht

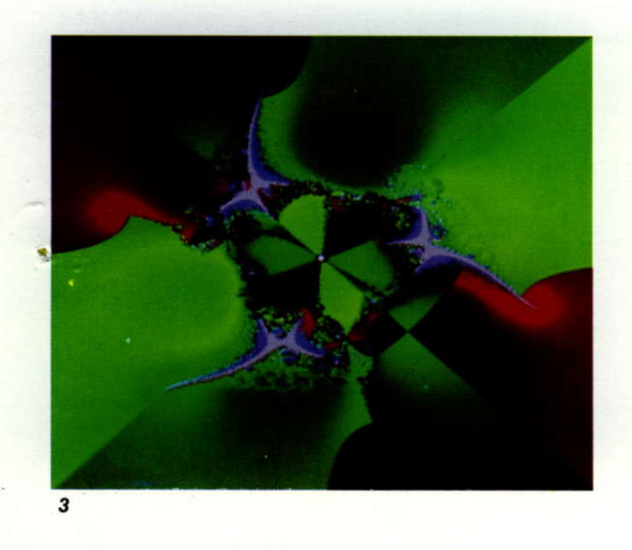
Um die Nutzung der Wechsel-Festplatte nicht auf den Tandon PAC 286 zu beschränken, wird mittlerweile als Zusatzausstattung für IBM-PCs und Kompatible das sogenannte Add-on-PAC angeboten. Es besteht aus der RLL-Controllerkarte, die in einen Slot des XT oder AT gesteckt wird und einem Beistellgefäß, welches je nach Ausbaustufe ein oder zwei Data Pacs aufnimmt und als Box neben den PC gestellt werden kann.

Zur Nachahmung empfohlen

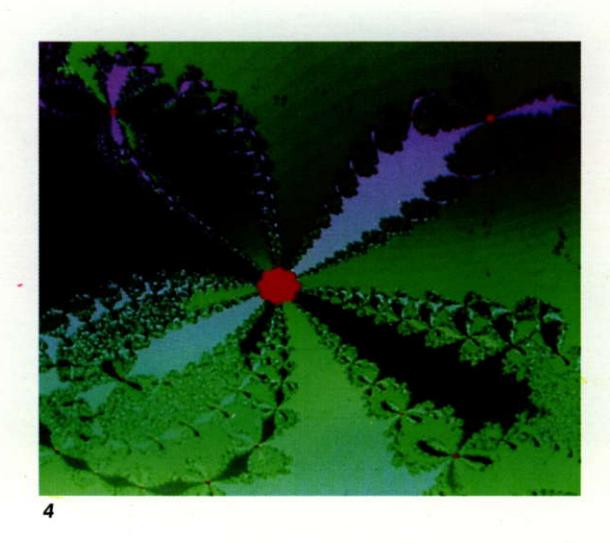
Einen dem Personal Data Pac vergleichbaren Gedanken realisiert inzwischen die Firma Victor mit dem sogenannten ADD-PAK am VPC III 286. Das ADD-PAK beinhaltet ebenfalls ein 3½-Zoll-Festplattenlaufwerk mit 30 MByte und kann nach Öffnen einer Klappe in der Frontseite des VPC montiert werden. Die Handhabung ist hier also nicht so anwenderfreundlich, daß sie mit der beim Data Pac oder gar einer Diskette vergleichbar wäre.
Zudem können im ADD-PAK herkömmliche Festplattenlaufwerke verwendet werden, die rauhen Transportbedingungen unter Umständen nicht gewachsen sind.
Sicher werden künftig noch mehr Hersteller ähnliche und verbesserte Lösungen anbieten; ob sich die Idee durchsetzen wird, bleibt abzuwarten – schließlich machen auch andere Datenträger, zum Beispiel optische Speicher, Bernoulli-Box, Plattenkassetten, von sich reden.

MP–We

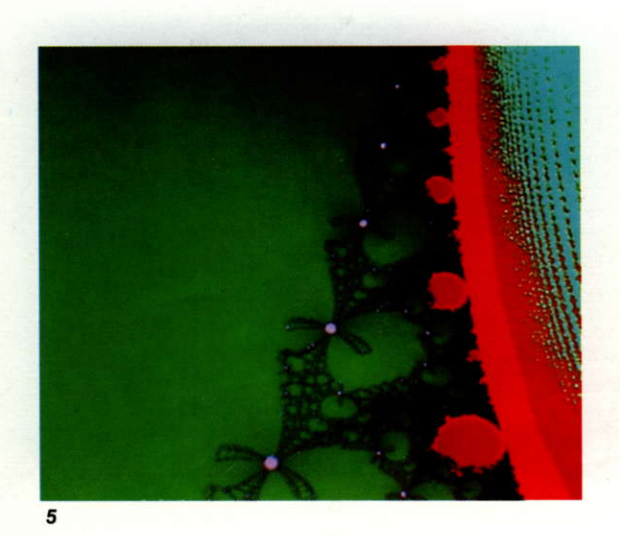

ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK
MP
Heft 9 · 1988
Mikroprozessortechnik
VEB Verlag Technik Berlin
ISSN 0232-2892
RISC
In diesem Heft:
Vom Fraktal zum Naturgesetz?
MS-DOS



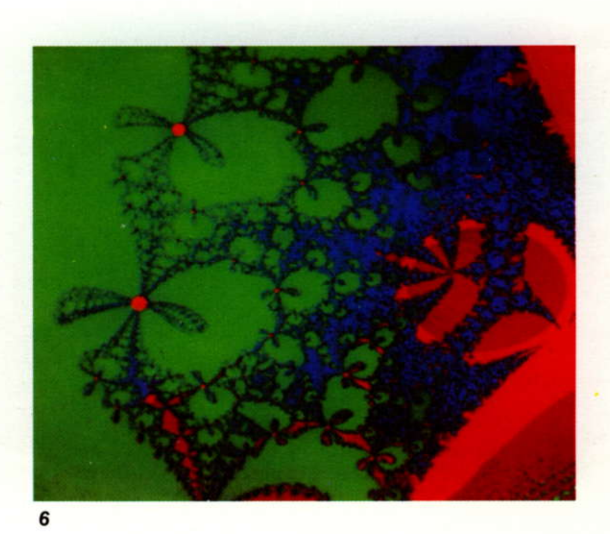
3



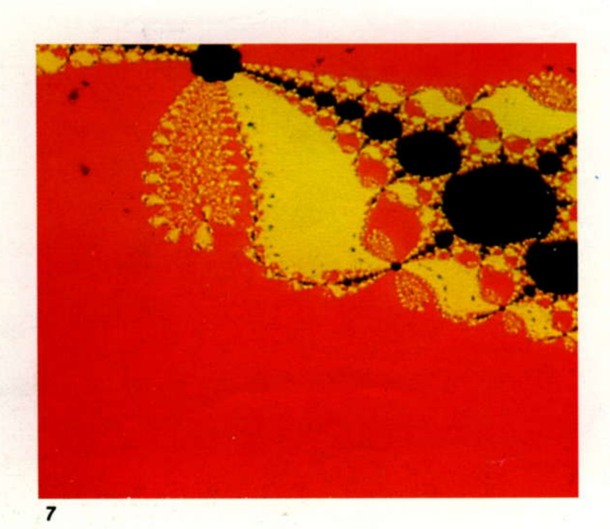
4



5



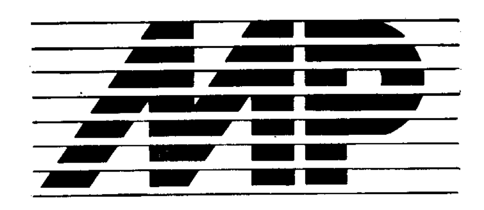
6



7

Lesen Sie dazu den Beitrag „Fraktale aus Polynomen"

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232-2892



Mikroprozessortechnik, Heft 9 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2870371); Herbert Hemke, Redakteur (Tel. 2870203); Sekretariat Tel. 2870381

Gestaltung Christina Bauer

Titelfotos Gabriele Buhren

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

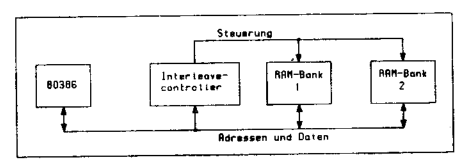
Redaktionsschluß: 18. Juli 1988

AN (EDV) 49837

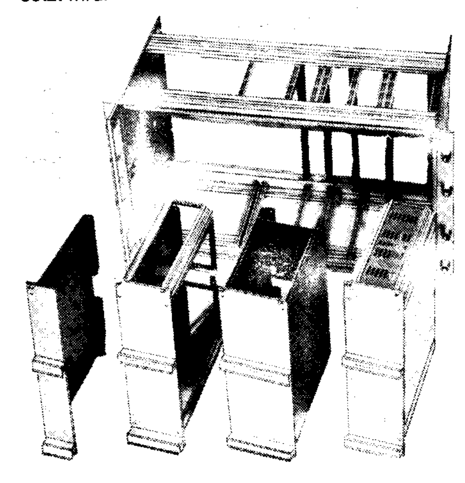
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

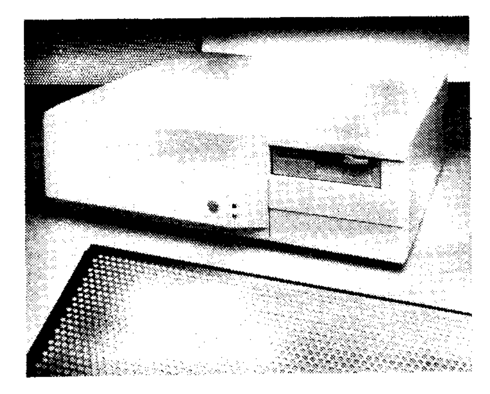
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tisku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Hà Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



Der dritte und letzte Teil unserer Artikelfolge „Moderne Mikrorechnersysteme" auf der **Seite 267** behandelt den 32-Bit-Prozessor 80386, der international zunehmend in PCs für hohe Ansprüche eingesetzt wird.



Auf der **Seite 275** finden Sie den Beitrag „Elektronische Baugruppen und mechanische Aufbausysteme für die Automatisierungstechnik". Er gibt einen Überblick über die internationale Standardisierung der Gefäßtechnik für elektronische Baugruppen sowie über die 19-Zoll-Einschubeinheiten aus dem Kombinat EAW.



Unser Bericht zur Hannover-Messe Industrie '88 auf der **Seite 283** erwähnt u. a. das Intel System 302, einen PC der AT-386-Klasse, der mit 25 MHz getaktet wird.

Vorschau
In Heft 10 finden Sie Beiträge zu folgenden Themen:
- Bildungscomputer A 5105
- dBASE III im Vergleich
- MODULA-2

Inhalt

Mehrlagenleiterplatten

Das Forschungszentrum des Werkzeugmaschinenbaues in Karl-Marx-Stadt hat mit der Fertigung von Mehrlagenleiterplatten begonnen. Sie ermöglichen eine weitere Erhöhung des Integrationsgrades und sind für die Verwirklichung wichtiger Forschungs- und Entwicklungsaufgaben im Industriezweig bestimmt. Mit ihrer Hilfe können die Entwicklungszeiten für neue Werkzeugmaschinensteuerungen wesentlich verkürzt werden. Seit 1981 bereits stellt die Karl-Marx-Städter Forschungsstätte in einem Kleinfertigungszentrum einfachere Typen von Leiterplatten her. Ihre Produktion stieg in den vergangenen sieben Jahren auf das Fünffache. Der DDR-Werkzeugmaschinenbau verfügt über mehrere Produktionsstätten für elektronische Baugruppen. Im Karl-Marx-Städter WMW-Forschungszentrum werden neben Leiterplatten auch Hybridschaltkreise gefertigt. Der Eigenbau von Elektronikbaugruppen trägt dazu bei, daß beispielsweise der Stammbetrieb des Fritz-Heckert-Kombinates in diesem Jahr mehr als 90 Prozent seiner Erzeugnisse mit mikroelektronischen Steuerungen und Antrieben ausstatten kann.

ADN

Computer als Mittel und Gegenstand psychologischer Forschung

Wissenschaftler der Berliner Humboldt-Universität stellten auf dem 7. Kongreß der Gesellschaft für Psychologie der DDR in Leipzig im Februar Modelle und Methoden vor, mit denen sich Spezialwissen für Expertensysteme erfassen läßt. Diese wissensgestützten Informationssysteme wären in vielen volkswirtschaftlichen Bereichen unter anderem als vorteilhafte Entscheidungshilfe einsetzbar. Schwierigkeiten bereitet es bisher, das dafür notwendige Faktenwissen herauszufiltern, zu beschreiben und in Software bereitzustellen. Am Beispiel des Wissensschatzes von Konstrukteuren schilderten die Berliner Fachleute den kombinierten Einsatz verschiedener Methoden vom Interview bis zum „lauten“ Denken, um die Kenntnisse von Spezialisten für die Eingabe in den Computer zu recherchieren.
Mit Möglichkeiten und Grenzen der computergestützten Psycho-Diagnostik setzten sich Gelehrte von der Leipziger Alma mater auseinander. Sie vertraten die Auffassung, daß die Rechentechnik in ihrer Wissenschaftsdisziplin nicht nur Routinearbeiten rationalisieren sollte, sondern sich ebenso für das Weiterentwickeln praktikabler Verfahren empfiehlt. Insbesondere für aussagekräftige Tests, Trainingsprogramme und Patientendateien. Computereinsatz stehe dabei keineswegs im Widerspruch zu Forderungen nach stärkerer Beachtung der Persönlichkeit der Patienten in der Diagnostik. Als Beispiel dafür diente das vorgestellte Programm PSYCHOSOFT für Lerntests, Experimente der allgemeinen Psychologie sowie Trainingsprogramme. Mit dem Programm können Konzentrationsübungen ausgeführt und Lese-Rechtschreib- sowie Rechenschwächen abgebaut werden. Die auch für Laien verständliche Softwarelösung setzt sich aus Modulen zusammen, die sowohl kompakt als auch einzeln nutzbar sind.

ADN

Elektronik für Nadeldrucker im Wärmetest

Einem Funktionstest im Wärmeschrank werden Elektronikbaugruppen für Nadeldrucker im Büromaschinenwerk „Ernst Thälmann“ Sömmerda vor der Endmontage unterzogen. Bei dieser mehrstündigen Qualitätskontrolle bewährt sich eine Rationalisierungslösung junger Facharbeiter und Technologen des Robotron-Betriebes. Die einzelnen Testplätze wurden mit einem Mikrorechner gekoppelt, der Temperaturen und Meßergebnisse überwacht und protokolliert. Durch seinen Einsatz lassen sich jetzt Leiterplatten, Netzteile, Speicher und weitere Baugruppen von 240 Druckern gleichzeitig prüfen. Auftretende Fehler werden genau lokalisiert; dafür entwickelten die jungen Neuerer ein spezielles Programm, das mehr als 20 verschiedene Testfunktionen realisiert. Durch Einführung der rechnergestützten Wärmeverbundprüfung können jährlich mehr als 5.000 Stunden Arbeitszeit eingespart werden. Mit ihren Rationalisierungslösungen tragen Jugendbrigaden und -forscherkollektive dazu bei, die Produktion von Druckern in diesem Jahr auf rund 127 Prozent zu erhöhen. Erstmals sollen den größten Robotron-Betrieb mehr als 100 000 dieser Ausgabegeräte für die CAD/CAM-Technologie verlassen.

ADN

Westliche Industriestaaten verstärken Druck auf Länder Asiens

Trotz erheblicher Zuwachsraten im Exportgeschäft in den ersten vier Monaten dieses Jahres mehren sich im südostasiatischen Stadtstaat Singapur Stimmen, die diese Entwicklung als unbeständig einschätzen. Im April wurden 22,9 Prozent mehr Erzeugnisse als im gleichen Monat des Vorjahres exportiert. Die verarbeitende Industrie war mit Computerprodukten und Fernmeldeeinrichtungen zu 70 Prozent an diesem Anstieg beteiligt. Der Vorsitzende des Handelsentwicklungsrates von Singapur, Alan Yeo, erklärte jedoch, sein Land werde diesen Trend nicht kontinuierlich fortsetzen können. Er machte dafür den wachsenden Protektionismus der USA, der EG-Staaten und Japans verantwortlich, wohin 60 Prozent aller Exporte Singapurs fließen.
Neben Singapur sind auch die anderen drei neuindustrialisierten Länder (NIC) Asiens – Südkorea, Taiwan und Hongkong – von protektionistischen Maßnahmen betroffen. Dazu gehören Tarife, die den Export von vornherein verhindern.
Mit Argwohn wurden in Singapur, Soul, Hongkong und Taibei die jüngsten Beratungen der 24 Mitgliedstaaten der Organisation für wirtschaftliche Zusammenarbeit und Entwicklung (OECD) in Paris beobachtet. Obwohl sich die Minister der führenden kapitalistischen Staaten eingehend mit der Wirtschaft der NIC beschäftigten und ihr hohe Konkurrenzfähigkeit bescheinigten, boten sie deren Vertretern keinen Platz am Tisch ihres exklusiven Klubs an. Statt dessen wurden weitergehende wirtschaftspolitische Forderungen an die NIC gerichtet.
Insbesondere die USA verlangen von den vier Staaten unter anderem verstärkte Bildung von Unternehmen mit ausländischer Beteiligung, unbegrenzten Profittransfer, die Einschränkung von Importbegrenzungen sowie eine weitere Öffnung für westliche Investitionen. USA-Finanzminister James A. Baker ging sogar soweit, den NIC die Schuld für Wirtschaftsprobleme der kapitalistischen Staaten zuzuschieben. Taiwan und Südkorea warf er vor, ihre Weigerung zur Öffnung gegenüber neuen Investitionen behindere Bemühungen zur Stabilisierung der Weltwirtschaft.

ADN

Ratgeber in Sachen Rechentechnik

Software für Betriebe und Einrichtungen ihres Kreises entwickeln Mitglieder der Wissenschafts- und Produktionsgemeinschaft (WPG) Mikroelektronik-Rechentechnik Dippoldiswalde. Die Fachleute arbeiten in der einzigen Gemeinschaft dieser Art im Bezirk Dresden unter anderem ein Programm zur rechnergestützten Ermittlung des Materialbedarfs für Gießereimaschinen aus. Es wird genutzt im VEB Gießerei- und Maschinenbau „Ferdinand Kunert“ Schmiedeberg. Die Wissenschafts- und Produktionsgemeinschaft mit ihrem Sitz an der Ingenieurschule für Lebensmittelindustrie Dippoldiswalde wurde vor einem Jahr gegründet, um Betrieben, Genossenschaften und Einrichtungen bei der Einführung und breiten Anwendung der Mikroelektronik und Rechentechnik zu helfen. Ihr gehören die beiden Ingenieurschulen Dippoldiswalde und Glashütte, der Rat des Kreises sowie sieben Betriebe des Territoriums an. Die Gemeinschaft entwickelt jedoch nicht nur betriebsspezifische Software. Seit Oktober vergangenen Jahres wurden rund 60 Werktätige in Lehrgängen mit der Arbeit an Computern vertraut gemacht. Auch kleinere Betriebe, die nicht Mitglied sind, informieren sich über Möglichkeiten der Anwendung dezentraler Rechentechnik. Als Vorhaben der Dippoldiswalder WPG – entstanden nach dem Vorbild einer derartigen Organisation in Haldensleben im Bezirk Magdeburg – steht die Wartung der Hardware für die Mitgliedsbetriebe auf dem Programm.

ADN

NEC entwickelte bisher schnellsten 16-Bit-Mikroprozessor

Das japanische Elektronikunternehmen NEC stellte Anfang April einen 16-Bit-Hochgeschwindigkeits-Mikroprozessor vor, der nach Angaben der japanischen Fachpresse die gegenwärtig schnellste 16-Bit-CPU der Welt ist. Der Prozessor, der mit 12,5 und 16 Megahertz angeboten wird, kann bis zu 2,8 Millionen Instruktionen in der Sekunde (MIPS) verarbeiten. Seine Geschwindigkeit erhält der neue Prozessor durch ein Hard-Kabel-System, das die Zeitfrequenz erhöht sowie die Teilung von Adressen- und Datenleitungen gewährleistet, was die Daten-Transmission verkürzt. Gleichzeitig verfügt der neue V33 als neunter Mikroprozessor der V-Serie von NEC im Vergleich zu seinem Vorgänger V30 über eine erweiterte Speicherkapazität.

ADN

Das Forschungsinstitut für Computer-Technologie in Peking

entwickelte ein Bildschirmsystem, mit dem chinesische Schriftzeichen, grafische Darstellungen sowie fremdsprachige Texte aufgezeichnet werden können.

Foto: ADN-ZB/HNA



Kleincomputerproduktion in Zusammenarbeit

Ein Kiewer Betrieb, der sich auf die Produktion von Computern des SKR spezialisiert, hat vor kurzem zusammen mit anderen sowjetischen Betrieben das Recht auf eigenständigen Verkauf seiner Erzeugnisse im Ausland erhalten. Drei RGW-Länder – DDR, Polen und CSSR – sind heute die größten Importeure von „Elektronmasch". Auf sie entfallen etwa 70 Prozent des Exports. Die Partner liefern an „Elektronmasch" Komponenten zur Komplettierung der Rechner, und der Betrieb verkauft an sie fertige Steuerrechnerkomplexe.
Im vergangenen Jahr wurde auch mit dem ungarischen Betrieb Videoton eine Produktionskooperation in Gang gesetzt. Im Rahmen des Komplexprogramms des wissenschaftlich-technischen Fortschritts der RGW-Länder bis zum Jahre 2000 wurde ein zeitweiliges sowjetisch-ungarisches Projektierungs- und Konstruktionsbüro für die Entwicklung neuer Typen von Banddruckern für Kleincomputer gegründet. Bereits im ersten Halbjahr beginnen die ungarischen Partner mit ihrer Produktion und liefern sie an „Elektronmasch". Mit der Lieferung einzelner Computerbaugruppen nach Kiew erhalten die ungarischen Freunde dann fertige Computerkomplexe.
Die vor kurzem bei der Kiewer Vereinigung gegründete Außenhandelsfirma ermöglicht schon heute, auf den Produktionsprozeß positiv Einfluß zu nehmen. Mit der neuen Arbeitsweise ist der Betrieb in der Lage, operativer auf die Wünsche der Partner in den RGW-Ländern zu reagieren und flexibler solche Probleme in Angriff zu nehmen wie die Preisbildung und die Organisation von Wartungsarbeiten.
Problematisch ist noch die Qualität der Rechentechnik, die in unseren Ländern produziert wird; sie entspricht zwar bei einigen Kennziffern dem Weltniveau, bleibt aber zurück was Zuverlässigkeit, Gewicht und Ausmaße sowie Energieverbrauch anbelangt.
Die Erfahrungen der vergangenen Zeit zeigen aber, daß der eingeschlagene Weg der Kooperation in Wissenschaft, Technik und Produktion mit den RGW-Ländern richtig ist.

APN/ADN-Sirota

Dialog

Computerviren

Seit geraumer Zeit bin ich bemüht, auf zwei Fragen eine Antwort zu finden:
Was ist unter einem Virus zu verstehen?
Wie kann man den Computer vor ihm schützen?
Für die Beantwortung dieser beiden Fragen oder einen entsprechenden Literaturhinweis wäre ich Ihnen sehr dankbar.

C. G., Bitterfeld

Die Ihnen bekannten biologischen Viren haben ein Äquivalent in der Computertechnik gefunden, sie haben sogar gemeinsame Merkmale in Funktion und Lebensweise. Ohne Ihre Fragen an dieser Stelle erschöpfend beantworten zu können, lassen sich Computerviren wie folgt charakterisieren:
– Ein Computervirus ist ein künstliches Gebilde.
– Es ist in jedem Fall ein Programm.
– In einem Computer ohne Verbindung mit der Außenwelt (Modem) ohne fremde Programme besteht keine Infektionsgefahr.
– Ein abgeschalteter Computer ist einem toten Organismus gleichzusetzen, er kann nicht infiziert werden.
– Virenprogramme dienen oder dienten bestimmten Zwecken, eine Verbreitung erfolgt beabsichtigt oder unbeabsichtigt.
– Computerviren können als reines Virus-Programm oder als virusinfiziertes beliebiges Programm verbreitet werden.
– Computerviren wurden geschaffen, um Programme oder Datensätze unter bestimmten Umständen zu kennzeichnen, zu manipulieren oder zu zerstören.
– Computerviren haben eine vom Programmierer vorbestimmte Inkubationszeit; auf einen Programmcode, eine Programmlaufzeit, ein Bitmuster oder eine bestimmte Datenmenge abgestimmt, beginnt der Virus sein Werk.
– Computerviren sind in allen Programmiersprachen erstellbar.
Die Hardware ist nur in wenigen Fällen zu beschädigen; so ist es z. B. möglich, das Diskettenlaufwerk auf eine Kopfposition zu bringen, aus der es nur noch die teilweise Demontage wieder zum Laufen bringt. Des weiteren können Disketten und Festplatten ohne Absicht des Bedieners sinnlos beschrieben oder gelöscht werden. Bei Druckvorgängen könnten solche Virenmanipulationen eingebaut werden, die irgendwo im Text die Papiervortriebsrichtung beim Seitenwechsel mehrfach umschalten – die Folge ist zumindest Papiersalat im Drucker...
Zum zweiten Teil der Frage: Ein absolut wirkendes Schutzprogramm ist nicht zu erstellen. In Abhängigkeit vom Typ Ihres Computers sollten Sie neue Programme aus nicht kommerziellen Quellen nicht mit wertvollen Programmen auf einem Speichermedium verwenden. Verwenden Sie nur Arbeitskopien dieser Programme. Entstehen beim Abarbeiten bekannter Programme plötzlich andere Ergebnisse, verlängert sich die Rechenzeit oder ist der verfügbare Speicherraum (RAM, RAMDISK, Diskette) in kurzer Zeit ohne Ihr Dazutun erschöpft, könnte ein Virus die Ursache sein. Manche Virusinfektionen täuschen auch einen Hardwaredefekt vor.
Eine Empfehlung vielleicht, bilden Sie die Quersumme eines Programmes oder eines Datensatzes und notieren Sie diese Zahlen extern auf Papier – nicht im Computer! Unter CP/M läuft das Programm CRC.COM und gibt Ihnen die Einzelergebnisse und die gesamte Quersumme des betreffenden Speichermediums. So ist ein unabsichtlich manipuliertes Programm immer mit dem „Original" vergleichbar!
Ein ausgeschalteter Computer ohne Stützspannung und ohne Festplatte bzw. Diskette, von der das Betriebssystem erst geladen wird, ist (sofern nicht beim Hersteller mit Viren infiziert) virenfrei. Das heißt, nach dem Einschalten hat man unter diesen Bedingungen den Computerurzustand.
Abschließend noch einige Literaturstellen, die über zentrale Bibliotheken zu beziehen sind:

/1/ Krabel, E.: Die Viren kommen. c't (1987) 4
/2/ Computerviren. 64er (1987) 7
/3/ Burger, R.: Das große Computer-Viren-Buch. Data Becker Düsseldorf 1987
/4/ Sperber, J.: Virusfieber, mc (1988) 7
/5/ Risch, H.-J.: Angst vor Computer-Viren; Koziel, T., Leister, G.: Die Viren sind da. c't (1988) 7

Intel will Entwicklungszeit beim 80 486 stark verkürzen

Als die Intel Corp. den Mikroprozessor 80 286 entwickelte, waren auf diesem Chip 120 000 Transistoren untergebracht, und die Entwicklungszeit betrug etwa dreieinhalb Jahre. Auf seinem Nachfolger, dem 80 386, waren schon mehr als zweimal so viele Transistoren untergebracht – genau 275 000 –, und seine Entwicklungszeit betrug etwas weniger als drei Jahre. Jetzt entwirft eine Gruppe von Entwicklern bei Intel den Mikroprozessor 80 486 mit einer Million Transistoren (siehe auch MP 4/88). Sie hoffen, daß sie den ersten Prototyp in weniger als zwei Jahren hergestellt haben.
„Wir haben gemerkt, daß man die Entwicklungszeit, die man vom Konzept bis zum fertigen Chip benötigt, ständig verringern kann", meinte Albert Yu, Vizepräsident der Gruppe für Mikrocomputerbauteile von Intel. Yu ist einer der Planer des Programms für den „52-Wochen-Chip". Es soll dazu dienen, die durchschnittliche Entwicklungszeit eines logischen Chips auf ein Jahr oder weniger zu verringern. Um die Entwicklungszeit zu verkürzen, hat Intel einige grundlegende Prinzipien von Waffen-, Auto- und Klimaanlagenherstellern übernommen, die schon seit langem um die Vorzüge wissen, neue Erzeugnisse aus Teilen von alten zu entwickeln. Bei Intel sind die „alten Teile" Schaltkreise und deren Kombinationen. Ein Chipdesigner hat vielleicht Wochen oder Monate benötigt, um ein Bus-Interface zu entwickeln, das die Kommunikation zwischen dem Prozessor und den peripheren Geräten steuert.
Im sogenannten „52-Wochen-Programm" sind alle wesentlichen Merkmale in einem Indexspeicher eines Computers der Entwicklergruppe zusammengefaßt. Mit einem Tastendruck kann ein bewährtes Bauelement des 80 386 aus dem Speicher geholt und auf dem Bildschirm in den 80 486 eingepaßt werden. Wenn notwendig, kann das Bauteil für seine neue Funktion modifiziert werden. Ein Vorteil der „elektronischen Bibliothek" besteht darin, daß sie dem Entwickler eine kurze Zusammenfassung der jeweiligen Rolle einer Schaltkreisgruppe bietet, so daß er nicht jeden Transistor in allen Einzelheiten zeichnen muß. Die Ingenieure können auf dem Bildschirm Gruppen von Bauelementen hin- und herschieben und herausfinden, wie sie am besten untergebracht werden können. Dieses Verfahren birgt jedoch auch Risiken in sich. Jede Gruppe von Schaltkreisen, die aufgenommen worden ist, muß fehlerfrei sein. Ein Fehler in einem Chip kann auf andere Schaltkreise übertragen werden. Deshalb muß jeder Schaltkreis außerordentlich genau geprüft werden, das heißt, ein Designer muß sich sicher sein, daß sich jede Verbindung zwischen den Bauteilen auf dem neuen Chip unter genauer Betrachtung der strengen Designregeln an der richtigen Stelle befindet.

ADN

OS/2 und Windows 386 von Siemens

Seit April liefert Siemens seine Version des neuen Betriebssystems OS/2 sowie Windows 386 aus; Windows 2.0 seit 1988.
Die Betriebssystemerweiterung Windows 386 ist eine weiterentwickelte Form der gewohnten grafischen Bedienoberfläche von Microsoft. Die Fenster können jetzt überlappen, sind in der Größe veränderbar und lassen sich auf dem Bildschirm verschieben. Die Bedienoberfläche entspricht der des Presentation Manager, also der Bedienoberfläche von OS/2. Der Anwender kann also heute schon mit einem System arbeiten, das von vielen als zukünftiger Standard bei der Bedienung von Personalcomputern angesehen wird.
Mit Windows 386 wird Multitasking-Betrieb ermöglicht. Die besonderen Eigenschaften des 80386-Prozessors werden derart genutzt, daß mehrere Anwendungen im 8086-Modus gleichzeitig auf dem 80386-Prozessor nachgebildet werden. Dadurch können herkömmliche MS-DOS-Programme gleichzeitig im Speicher gehalten werden und auch ablaufen. Der Benutzer, der häufig zwischen Programmen wechselt, muß so die Programme nicht jedesmal neu laden. Man kann z. B. im Vordergrund eine Windows-Applikation ablaufen lassen, während im Hintergrund, sichtbar in einem Fenster, z. B. Multiplantabellen bearbeitet werden. Ein Textaustausch über die Zwischenablage ist jetzt nicht nur mit Windows-Programmen, sondern auch mit normalen MS-DOS-Anwendungen möglich. Windows 386 kann damit in der Migrationsphase vom MS-DOS zum OS/2 als Übergangsprodukt betrachtet werden.

MP

In eigener Sache

Unser Verlag sucht ab 1989 Beteiligung an Kinderferienlager für etwa 20 Kinder. Betreuer werden gestellt.
Wir bitten, Angebote an die Redaktion zu richten.

Fraktale aus Polynomen

Gabriele Buhren, Berlin

Bisher veröffentlichte fraktale Motive beruhen auf Varianten zufällig gefundener Rekursionsgleichungen oder entsprechen Fixpunkt-Lösungen ohne Bifurkationen (Lösungsaufspaltungen). Das hier geschilderte Verfahren ist neu, ebenso die Art der Bilder. Es kann jede Komplexe Funktion $f(z) = 0$ benutzt werden, die ein eindeutiges $f'(z)$ besitzt.

Mathematische Methode mit Beispielen

Ein bekanntes Fraktal

Jedes beliebige Polynom $f(z) = 0$ kann durch Iteration mit dem *Newton*-Verfahren

$$z_{n+1} = z_n - \frac{f(z_n)}{f'(z_n)} \qquad (*)$$

gelöst werden, auch wenn $z = x + iy$ eine komplexe Größe ist. Bei Mehrfachlösungen hängt das konkrete Einzugsgebiet (als Farbe im Bild) von der Wahl der Anfangswerte x_a und y_a ab (Koordinaten eines Bildpunktes). Für die drei Fixpunkte der Gleichung

$$z^3 + 1 = 0 \qquad (**)$$

ergibt sich das bekannte Fraktal, an dessen Linien immer alle drei Gebiete aneinanderstoßen. Nennen wir es den dreizackigen „Einfachstern".

Dimensionserweiterung

Setzt man jetzt für z eine bestimmte hyperkomplexe Zahl ein (siehe Tafel 1) und löst (**) mit (*) für alle Komponenten von z, dann ergibt sich mit dem Ansatz (1) ein „Doppelstern" (6zackig) aus 9 Farben, mit dem Ansatz (2) ein „Dreifachstern" mit 27 Farben (Bild 13, 3. US). Die hyperkomplexen Definitionen Quaternion (doppeltkomplexe Zahl) oder Oktave (doppeltes Quaternion) /3/ verändern den Einfachstern im Prinzip nicht und führen auch nicht zu dem im nächsten Abschnitt beschriebenen Effekt. Wohl aber Ansatz (3) und ähnliche Definitionen wie die Ansätze (4)–(6). Bei (3) ist zu beachten, daß die Anfangswerte der einzelnen Komponenten niemals gleich sein dürfen.

Tafel 1

$i^2 = -1$

Ansatz (1):
4dimensionale hyperkomplexe Zahl

$$Z1 = (x + iy) + j(p + it) \quad \text{mit} \quad j^2 = -i$$
$$= K0 + j\,K1 \quad \text{wobei}$$
$$Z1^* = K0 - j\,K1 \quad \text{und}$$
$$Z1 \cdot Z1^* = K0^2 + i \cdot K1^2$$

Mit Z^* erfolgt der erste Schritt beim Reellmachen des Nenners von (*). Der zweite ist die normale Multiplikation mit der konjugiert-komplexen Zahl.

Ansatz (2):
6dimensionale hyperkomplexe Zahl

$$Z2 = (x + iy) + j(p + it) + j^2(k + il)$$
$$= K0 + j\,K1 + j^2\,K2 \quad \text{mit} \quad j^3 = -i$$
$$Z2^* = (K0^2 + i \cdot K1 \cdot K2) + j(-i \cdot K2^2 - K0 \cdot K1) + j^2(K1^2 - K0 \cdot K2)$$
$$Z2 \cdot Z2^* = (K0^3 - K2^3) + i(3 \cdot K0 \cdot K1 \cdot K2 - K1^3)$$

(Prof. Peschel)

Ansatz (3):
6dimensionale hyperkomplexe Zahl

$$Z3 = (x + iy) + j(p + it) + s(k + il)$$
$$j^2 = -1 - i$$
$$s^2 = 1 + i$$
$$js = -sj = 1 - i$$
$$= K0 + j\,K1 + s\,K2$$
$$Z3^* = K0 - j\,K1 - s\,K2$$
$$Z3 \cdot Z3^* = (K0^2 + K1^2 - K2^2) + i(K1^2 - K2^2)$$

Ansatz (4):
8dimensionale hyperkomplexe Zahl

$$Z4 = (K0 + j\,K1) + s(K2 + j\,K3)$$
$$j^2 = -i$$
$$s^2 = -j$$

Ansatz (5):
8dimensionale hyperkomplexe Zahl

$$Z5 = Z4 \quad \text{mit} \quad j^2 = -1 - i, \; s^2 = -1 - i - j - ji$$

Ansatz (6):
24dimensionale hyperkomplexe Zahl

$$Z6 = Z2 + s(\ldots) + s^2(\ldots) + s^3(\ldots)$$

$((\ldots)$ ist Größe wie Z2$)$ $\quad s^4 = -j$

Rückprojektion durch Einschränkung der Freiheitsgrade

Wenn nun Gleichung (*) in verstümmelter Form benutzt wird, indem man die Iteration einer oder mehrerer Variablen unterläßt, sie als Konstanten behandelt, entfernt man sich vom ehemaligen f(z). Man bearbeitet für jeden Bildpunkt nun eine zwar benachbarte, aber neue nichtlineare Gleichung, die im allgemeinen nicht mehr analytisch darstellbar ist. Das vorher durch (*) abgeschlossene (konservative) System des „Sterns" wird geöffnet, erhält Restriktionen durch das Nullsetzen oder Verändern von z. B. dp und dt (s. Kommentarklammern Zeile 60). Jeder Eingriff solcherart muß als Verlust bzw. Zuwachs bezeichnet werden, vergleichbar mit Dissipation.

Es entstehen Bilder neuartiger Struktur. Für p und t um ±2 endet das Gebilde, während der Stern bis Unendlich reicht. Eine Entsprechung wäre das Mandelbrotsche Apfelmännchen, jedoch ohne Ähnlichkeiten. In der Nähe des Bildmittelpunktes bei sehr kleinen p,t-Werten entsteht der Einfachstern. Je mehr man nach außen wandert, desto „verbeulter" werden seine Glieder. Dann entstehen sprunghaft völlig neue Fraktale. Die alten Fixpunkte sind zerfallen, neue erschienen, und die Gebiete mit zyklischen Lösungen erinnern an „Wirbelsäulen" und „Insektenflügel". „Kakteenkugeln" mit blumiger Feinstruktur tauchen auf. Durch eine entsprechende Farbkodierung lassen sich somit Welten darstellen, die der lebendigen Welt nicht unähnlich sind. Bisher konnten nur wenige Proben aus dieser unentdeckten Vielfalt entnommen werden. Es bleibt dem Leser überlassen, das spannende Spiel fortzusetzen. Das Phänomen der Fraktale ist noch relativ ungeklärt. Falls man zufällig einen hyperkomplexen Ansatz findet, der für ein passendes f(z) am echtesten reale Objekte (z. B. Eisblumen, Schaum, Organismen) modelliert, sollten sowohl die Biologen als auch die Physiker nach Anwendungen dieser hyperkomplexen Größen Ausschau halten.

Rechentechnische Realisierung

Da die Bilder auf einem Robotron-BVS A 6472 visualisiert werden konnten, mußten sie optimal aus 512 × 512 Bildpunkten zu je 8 Bit bestehen. Mit einem speziellen Programm wurde aus etwa 200 Varianten die günstigste Farbtabelle gewählt, um die Grauwerte in Falschfarben darzustellen. Zum Rechnen der Bilder kamen SKR-Rechner zum Einsatz (SM4, SM14), wobei die Rechenzeiten zwischen 3 und 20 Stunden lagen. Übertragen wurden die fertigen Bilder per Magnetband, danach vom Farbmonitor abfotografiert mit einer Practica (Teleobjektiv, UT 18,1/8 s).

Prinzipiell ist die Nutzung jedes mit PASCAL programmierbaren Computers möglich, wobei jedoch sehr lange Rechenzeiten auftreten können. In diesem Fall sind kleinere Bilder zu empfehlen (überall Austausch der Zahl 512). Die spätere Visualisierung muß gesondert an die vorhandene Technik angepaßt werden.

Programmbeschreibung

Unter Austausch der Zeilen 38–40, 47–61 und 84 wird das Programm (PASCAL, siehe Bild 1) auch zur bildmäßigen Simulation beliebiger dynamischer Systeme genutzt. Es kann dadurch für viele Anwendungszwecke brauchbar sein.

Tafel 2 Bildparameter (Bild = komplexe Ebene mit Koordinaten P + iT)

Nr.	Umschlagseite	Typ	BREITE	VSX	VSY	Anfangswerte		K	Ansatz
1	1	B	4.0	0.0	0.0	Xa = 0.7937005	Ya = −Xa	1 + i + j + ji	1
2	1	B	4.0	0.0	0.0	Xa = P	Ya = T	1 + j	1
3	2	A	4.0	0.0	0.0				
4	2	B	0.5	− 0.8	0.0				
5	2	A	0.5	− 1.0	1.0	Xa = P	Ya = T	1 + i	1
6	2	A	0.5	− 1.0	0.5				
7	2	A	0.1	− 1.225	− 0.06				
8	3	A	0.64	− 0.9219	− 0.9531	Xa = −1.0	Ya = 0.0	1	1
9	3	A	4.0	0.0	0.0				
10	3	B	0.5	− 0.25	0.75	Xa = 0.7937005	Ya = − Xa	1 + i	1
11	3	A	0.5	− 1.12	0.25				
12	3	A	0.1	− 1.12	0.25				
13	3	B	4.0	0.0	0.0	Xa = Pa = Ka = P	Ya = Ta =La = T	$1 + i + j + j^2 + ji + j^2i$	2

Bildkodierung

Es werden gleichzeitig zwei Bilder (A und B) zeilenweise auf Platte geschrieben (Zeile 89). Sie müssen zusammen betrachtet werden, wenn man das Lösungsverhalten auswerten will. Zur Kodierung stehen die Grauwerte 0 bis 225 zur Verfügung.

Im A-Bild wird für Fixpunkte als Grundfarbe der Quadrant abgebildet, in dem die Lösung liegt. Das hat den Nachteil, daß mehrere Attraktoren (Anziehungspunkte) im gleichen Quadranten nicht getrennt werden, und den

```
 1   PROGRAM HYP4;
 2     (*   4-DIM ZU ANSATZ (1)    , Z**3+K=0   *)
 3   CONST    BREITE = 4.0;                  (* GESAMTE BILDBREITE, BILD QUADRATISCH *)
 4            VSX = 0.0;  VSY = 0.0;       (* VERSCHIEBUNG DES BILDMITTELPUNKTES X,Y *)
 5            GENAU = 0.000001;                           (* ABBRUCH - GENAUIGKEIT *)
 6            K1 = 1.0; KI = 1.0; KJ = 0.0; KJI = 0.0; (* 4-DIM.KONSTANTE K IN F(X) *)
 7   LABEL    1,2;
 8   TYPE     ZEILE =  ARRAY[1..256] of INTEGER;
 9   VAR      ANZ,IIX,IX,IY,IYA,IYE,IG,IG1,L,ANZF : INTEGER;
10            SW,PANF,TANF,P,T,X,Y,XALT,YALT,DD,XX1,YY1,G1 : REAL;
11            P2,T2,PT6,Q3,P3,T3,PQT3,PTQ3,P6,T6,FSB,FDX,FDY,FDP,FDT : REAL;
12            A,B,C,D,E,F,G,H,X2,Y2,X3,Y3,QD,XY6,GA,GB,GC,GD,GE,GF : REAL; .
13            IZ,IZ1 : ZEILE;
14            FILNAM,FILN : PACKED ARRAY[1..8] OF CHAR;
15            F1,FF : FILE OF ZEILE;
16            W : BOOLEAN;
17
18   FUNCTION GRAU(K:INTEGER):INTEGER;
19   VAR  KK,GR : INTEGER;
20   BEGIN GR := ANZ+K; KK := K+50;
21    IF GR > KK   THEN  GR := KK;     GRAU := GR;
22   END;
23
24   BEGIN                                                    (* HAUPTPROGRAMM *)
25    WRITE(' Name der Datei : ');READLN(FILNAM); REWRITE(F1,FILNAM);
26    WRITE(' Zweiter Name   : ');READLN(FILN); REWRITE(FF,FILN);
27    WRITE(' ab Zeile : '); READLN(IYA); IY:=IYA-1; W:=TRUE;
28    WRITE(' bis Zeile: '); READLN(IYE); IYA:=(IYE-IYA+1)DIV 3; IIX:=0; IX:=0;
29    SW := BREITE/512.0;
30    PANF := -BREITE/2.0-SW+VSX;
31    TANF :=  BREITE/2.0+SW+VSY; TANF:=TANF-IY*SW;
32    REPEAT                                           (* ANF. ZEILEN - SCHLEIFE *)
33    TANF:=TANF-SW;   IY:=IY+1;
34     REPEAT                                          (* ANF. PUNKTE - SCHLEIFE *)
35     PANF:=PANF+SW; IX:=IX+1; ANZ:=0; ANZF:=0; L:=0;
36     IF W = TRUE THEN BEGIN IIX:=IIX+1; W:=FALSE; END
37                ELSE W:=TRUE;
38     P:=PANF; T:=TANF; X:=PANF; Y:=TANF;
39         PT6:=6.0*P*T; P2:=P*P; T2:=T*T; Q3:=3.0*(P2-T2); P3:=P*P2; T3:=T*T2;
40         PQT3:=3.0*P2*T; PTQ3:=3.0*P*T2; P6:=6.0*P; T6:=6.0*T;
41
42     1: ANZ:=ANZ+1; ANZF:=ANZF+1;                 (* ANF. SCHLEIFE FUER ITERATION *)
43        IF ANZ > 400 THEN IG:=10                                      (* ABBRUCH *)
44        ELSE
45        BEGIN                                                  (* NEUE ITERATION *)
46        XALT:=X; YALT:=Y;
47    X2:=X*X; Y2:=Y*Y; X3:=X*X2; Y3:=Y*Y2; QD:=(X2-Y2)*3.0; XY6:=K*Y*6.0;
48
49    A:=X3-3.0*X*Y2+Y*Q3+X*PT6+K1;   B:=3.0*Y*X2-Y3-X*Q3+Y*PT6+KI;
50    C:=P*QD-XY6*T+PQT3-T3+KJ;       D:=T*QD+XY6*P+PTQ3-P3+KJI;
51    E:=QD+PT6;      F:=XY6-Q3;      G:=X*P6-Y*T6;  H:=Y*P6+X*T6;
52    GA:=A*E-B*F-C*H-D*G;            GB:=A*F+B*E+C*G-D*H;
53    (*GC:=C*E-F*D-A*G+B*H;          GD:=E*D+F*C-A*H-B*G;*)
54    GE:=E*E-F*F-2.0*G*H;            GF:=G*G-H*H+2.0*E*F;
55    FSB:=GE*GE+GF*GF;
56     IF FSB < 1.0E-10 THEN BEGIN  IG:=1; IG1:=1; GOTO 2; END          (* ABBRUCH *)
57      ELSE
58      BEGIN   FDX:=(GA*GE+GB*GF)/FSB;   FDY:=(GB*GE-GA*GF)/FSB;
59              (*FDP:=(GC*GE+GD*GF)/FSB; FDT:=(GD*GE-GC*GF)/FSB;*)
60              X:=X-FDX;  Y:=Y-FDY;  (*P:=P-FDP;  T=T-FDT;*)
61       END;
62        IF(ABS(X)+ABS(Y))>200.0 THEN BEGIN IG:=255; IG1:=255; GOTO 2;  END (*ABB.*)
63         ELSE
64         BEGIN                                               (* ANF. ATTRAKTOR *)
65          IF ANZF < 50 THEN
66               BEGIN                                         (* ANF. FIXPUNKT *)
67               DD:=ABS(XALT-X)+ABS(YALT-Y); IF DD > GENAU THEN GOTO 1;
68                IF X > 0.0 THEN
69                  IF Y < 0.0 THEN  IG:= GRAU(200)      ELSE IG:= GRAU(150)
70                ELSE IF Y >= 0.0 THEN  IG:= GRAU(50)   ELSE IG:= GRAU(100);
71                END                                         (* FIXPUNKT END *)
72          ELSE
73              BEGIN                                       (* ANF. ZYKLUSZAHL *)
74              IF ( L > 0 )  AND ( L < 16 )  THEN
75                  BEGIN
76                  L:=L+1; DD:=ABS(XX1-X)+ABS(YY1-Y); IF DD > GENAU THEN GOTO 1;
77                   IF L = 2 THEN BEGIN  ANZF:=1; GOTO 1; END
78                           ELSE  IG:=19+L;    (* ZYKLUS GEFUNDEN ! *)
79                  END
80              ELSE  BEGIN XX1:=X; YY1:=Y; L:=1; GOTO 1; END;
81              END;                                        (* ZYKLUSZAHL END *)
82         END;                                              (* ATTRAKTOR END *)
83        END;                                          (* NEUE ITERATION END *)
84      G1:= ABS(X*0.7+Y)*127.0; IF G1 > 254.0 THEN G1:=254.0;  IG1:= ROUND(G1);
85   2:
86       IF W = TRUE THEN BEGIN IG1:=IG1*256+IZ1[IIX]; IG:=IG*256+IZ[IIX] END;
87      IZ[IIX]:=IG;    IZ1[IIX]:=IG1;                    (* 2 BYTE = 1 INTEGER *)
88     UNTIL  IX >= 512;                            (* PUNKTE - SCHLEIFE END *)
89     WRITE(F1,IZ);    WRITE(FF,IZ1);
90     WRITELN('  NEUE ZEILE BEGONNEN : ',(IY+1):3,'      (BIS ZEILE ',IYE:3,')');
91     PANF:= -BREITE/2.0-SW+VSX; IX:=0; IIX:=0; W:=TRUE;
92     IF ( IY = (IYE-2*IYA)) OR ( IY = (IYE-IYA)) THEN
93      BEGIN  CLOSE(F1); CLOSE(FF); REWRITE(F1,FILNAM); REWRITE(FF,FILN);
94      WRITELN('    NEUE FILES ! '); END;
95    UNTIL  IY >= IYE ;                          (*  ZEILEN - SCHLEIFE END *)
96    WRITELN('                PROGRAMM BEENDET');
97   END.                                              (*  HAUPTPROGRAMM END *)
```

```
(*  6-DIM. ZU ANSATZ(2) ---   P,T,K,L  FEST   *)
 .......     VEKTOR = ARRAY[1..6] of REAL;   .......
  ........     FKK,FZ,F,FS,FM      : VEKTOR;   .......
 (*  F = F(Z),  FS = F'(Z),  FZ = Z,  FKK = F*   *)
              ..........
 PROCEDURE MULTI(KF,G:VEKTOR;VAR FM:VEKTOR;M:INTEGER);
 LABEL 5;
 BEGIN
 FM[1]:=G[1]*KF[1]-G[2]*KF[2]+G[6]*KF[3]+G[5]*KF[4]+G[4]*KF[5]+G[3]*KF[6];
 FM[2]:=G[2]*KF[1]+G[1]*KF[2]-G[5]*KF[3]+G[6]*KF[4]-G[3]*KF[5]+G[4]*KF[6];
 IF M = 2 THEN  GOTO 5;
   FM[3]:=G[3]*KF[1]-G[4]*KF[2]+G[1]*KF[3]-G[2]*KF[4]+G[6]*KF[5]+G[5]*KF[6];
   FM[4]:=G[4]*KF[1]+G[3]*KF[2]+G[2]*KF[3]+G[1]*KF[4]-G[5]*KF[5]+G[6]*KF[6];
   FM[5]:=G[5]*KF[1]-G[6]*KF[2]+G[3]*KF[3]-G[4]*KF[4]+G[1]*KF[5]-G[2]*KF[6];
   FM[6]:=G[6]*KF[1]+G[5]*KF[2]+G[4]*KF[3]+G[3]*KF[4]+G[2]*KF[5]+G[1]*KF[6];
 5:            END;
               .............
 .......    GRZ:=1.0E15;  ......
 .......    P:=PANF; T:=TANF; X:=PANF; Y:=TANF; KK:=PANF; LL:=TANF;    ..........
FZ[1]:=X;  FZ[2]:=Y;  FZ[3]:=P;  FZ[4]:=T;  FZ[5]:=KK;  FZ[6]:=LL;
F[1]:=X*X-Y*Y+2.0*(P*LL+T*KK);  F[2]:=2.0*(X*Y-P*KK+T*LL);
F[3]:=2.0*(X*P-Y*T+KK*LL);      F[4]:=2.0*(X*T+Y*P)-KK*KK+LL*LL;
F[5]:=2.0*(X*KK-Y*LL)+P*P-T*T;  F[6]:=2.0*(Y*KK+X*LL+P*T);
          FOR I:=1 TO 6 DO            FS[I]:=3.0*F[I];
 MULTI(F,FZ,FM,6);
          FOR I:=1 TO 6 DO            F[I]:=FM[I]+1.0;
FKK[1]:=FS[1]*FS[1]-FS[2]*FS[2]-FS[3]*FS[6]-FS[4]*FS[5];
FKK[4]:=FS[6]*FS[6]-FS[5]*FS[5]-FS[1]*FS[4]-FS[2]*FS[3];
FKK[5]:=FS[3]*FS[3]-FS[4]*FS[4]-FS[1]*FS[5]+FS[2]*FS[6];
FKK[2]:=2.0*FS[1]*FS[2]+FS[3]*FS[5]-FS[4]*FS[6];
FKK[3]:=2.0*FS[5]*FS[6]-FS[1]*FS[3]+FS[2]*FS[4];
FKK[6]:=2.0*FS[3]*FS[4]-FS[1]*FS[6]-FS[2]*FS[5];
 MULTI(FS,FKK,FM,2);      ES:=FM[1];  GS:=FM[2];
 IF (ABS(ES) > GRZ) OR (ABS(GS) > GRZ) THEN
      BEGIN  IG:=254; IG1:=254; GOTO 2; END;
 MULTI( F,FKK,FM,2);      FSB:=ES*ES+GS*GS;    (*  6 STATT 2 FUER STERN !  *)
 IF FSB < 1.0E-10 THEN BEGIN  IG:=0; IG1:=0; GOTO 2; END;
 IF (ABS(FM[1]) > GRZ) OR (ABS(FM[2]) > GRZ) THEN
      BEGIN  IG:=254; IG1:=254; GOTO 2; END;
 FDX:=(FM[1]*ES+FM[2]*GS)/FSB;    FDY:=(FM[2]*ES-FM[1]*GS)/FSB;
(*FDP:=(FM[3]*ES+FM[4]*GS)/FSB;  FDT:=(FM[4]*ES-FM[3]*GS)/FSB;
 FDK:=(FM[5]*ES+FM[6]*GS)/FSB;    FDL:=(FM[6]*ES-FM[5]*GS)/FSB;*)
          X:=X-FDX;     Y:=Y-FDY;
          (*P:=P-FDP;  T:=T-FDT;  KK:=KK-FDK;  LL:=LL-FDL;*)
```

```
(*   6-DIM NACH ANSATZ (3)    *)
       .......
 PROCEDURE MULTI(KF,G:VEKTOR;VAR FM:VEKTOR;M:INTEGER);
 LABEL 5;
 VAR  H1,H2,H3,H4,H5,H6,H7,H8  : REAL;
 BEGIN
 H1:=G[5]+G[6]; H2:=G[3]-G[4]; H3:=G[5]-G[6]; H4:=-G[3]-G[4];
 H5:=H1+H2;     H6:=H3+H4;     H7:=H4-H3;     H8:=H1-H2;
 FM[1]:=KF[1]*G[1]-KF[2]*G[2]+KF[3]*H5+KF[4]*H6+KF[5]*H7+KF[6]*H8;
 FM[2]:=KF[1]*G[2]+KF[2]*G[1]-KF[3]*H6+KF[4]*H5-KF[5]*H8+KF[6]*H7;
 IF M = 2 THEN  GOTO 5;
  FM[3]:=KF[1]*G[3]-KF[2]*G[4]+KF[3]*G[1]-KF[4]*G[2];
  FM[4]:=KF[1]*G[4]+KF[2]*G[3]+KF[3]*G[2]+KF[4]*G[1];
  FM[5]:=KF[1]*G[5]-KF[2]*G[6]+KF[5]*G[1]-KF[6]*G[2];
  FM[6]:=KF[1]*G[6]+KF[2]*G[5]+KF[5]*G[2]+KF[6]*G[1];
 5:           END;
              ...........
FZ[1]:=X;  FZ[2]:=Y;  FZ[3]:=P;  FZ[4]:=T;  FZ[5]:=KK;  FZ[6]:=LL;
F[3]:=P*P-T*T-KK*KK+LL*LL;  F[4]:=2.0*(KK*LL-P*T);
F[1]:=X*X-Y*Y+F[3]+F[4];    F[2]:=F[3]+2.0*X*Y-F[4];
F[3]:=2.0*(X*P-Y*T);    F[4]:=2.0*(X*T+Y*P);
F[5]:=2.0*(X*KK-Y*LL); F[6]:=2.0*(X*LL+Y*KK);
          FOR I:=1 TO 6 DO           FS[I]:=3.0*F[I];
 MULTI(F,FZ,FM,6);
          FOR I:=1 TO 6 DO           F[I]:=FM[I]+1.0;
FKK[1]:=FS[1]; FKK[2]:=FS[2]; FOR I:=3 TO 6 DO FKK[I]:=-FS[I];
 MULTI(FS,FKK,FM,2);      ES:=FM[1];  GS:=FM[2];
 MULTI( F,FKK,FM,6);      FSB:=ES*ES+GS*GS;
 IF FSB < 1.0E-10 THEN BEGIN  IG:=0; IG1:=0; GOTO 2; END
 ELSE   BEGIN
FDX:=(FM[1]*ES+FM[2]*GS)/FSB;    FDY:=(FM[2]*ES-FM[1]*GS)/FSB;
FDP:=(FM[3]*ES+FM[4]*GS)/FSB;    FDT:=(FM[4]*ES-FM[3]*GS)/FSB;
FDK:=(FM[5]*ES+FM[6]*GS)/FSB;    FDL:=(FM[6]*ES-FM[5]*GS)/FSB;
          X:=X-FDX;  Y:=Y-FDY;
          P:=P-FDP;  T:=T-FDT;  KK:=KK-FDK;  LL:=LL-FDL;
        END;
```

Bild 2 Programmausschnitte für die Ansätze 2 und 3
Multiplikation der 6dimensionalen Arrays FM = KF · G

◀ ***Bild 1 Programm zu Bild 3 (2. Umschlagseite)***

Vorteil, das Nullwerden einer Komponente sehen zu können. Der Farbwechsel mit einer glatten Linie hat allerdings nichts mit den Fraktalen zu tun. Um zusätzlich die für eine vorgegebene Genauigkeit benötigte Iterationszahl abzubilden, wird auf den Grundgrauwert die Anzahl addiert (beschränkt bis 50). Nun sind alle Einzugsgebiete doch noch zu trennen, weil dort die Konvergenz stark fällt. Zufällig wurden auf diese Weise die „Netze" und „Flügel" sichtbar, die aus sehr schmalen Bifurkationen des Hintergrundes bestehen (kleiner 10^{-6}). Größere Bifurkationen kann das Programm trennen und konkret in Grauwerte des A-Bildes umsetzen. Da wäre Grauwert 22 ein Zweierzyklus, 23 ein Dreierzyklus usw. bis 35.

Das B-Bild trennt exakt alle Attraktoren voneinander. Es sieht „sauberer" aus, berücksichtigt aber weder Konvergenz noch Bifurkationsverhalten. Man erkennt jetzt, zu welchem Attraktor die in Bild A herausgehobenen Zyklen gehören. Die Kodierung erfolgt durch asymmetrische Wichtung des letzten x-y-Paares (Zeile 84), um Fehler bei häufigen symmetrischen Lösungen auszuschalten. Für beide Bilder gleich markiert sind die Abbruchbedingungen bei Divergenz, Division durch Null und Überschreiten von 400 Iterationen.

Eingaben

Wegen der relativ langen Rechenzeiten erwies es sich als günstig, das Bild abschnittsweise zu erzeugen, d. h. mehrmals denselben *Task* zu starten. Im Dialog gibt man zuerst die beiden File-Namen ein (z. B. H1.A bzw. H1.B), dann von welcher Zeile bis zu welcher Zeile gerechnet werden soll. Diesen Abschnitt teilt das Programm als Zwischensicherung in drei File-Versionen auf. Die fertigen Einzelteile lassen sich problemlos aneinanderhängen. Zufällig stimmt die Blocklänge der SKR-Rechner mit der Bild-Zeilenlänge überein (512 Byte), so daß die Blockzahl als Zeilenanzahl zu lesen ist. Dadurch kann man ohne Notizen bei der richtigen Zeile fortsetzen.

Um Eingabefehler zu vermeiden, sind die eigentlichen Bildparameter (siehe Tafel 2) im Programm zu ändern (Zeilen 3 bis 6 und 38). Zum Beispiel ergibt sich schon eine neue Gleichung durch Änderung der K-Komponenten ($z^3 + K = 0$). Für neue quadratische Bildausschnitte ändert man die Parameter BREITE, VSX und VSY, die sich auf die Gesamtbildbreite und den neuen Bildmittelpunkt beziehen. Beispielsweise sind die Bilder 4 bis 6 (2. Umschlagseite) Details des Bildes 3 (2. US) und die Bilder 10 bis 12 (3. US) Details des Bildes 9 (3. US). Wenn man als Anfangswerte für alle x und y eine der drei Lösungen bei p = t = 0 wählt (Zeile 38), erhält das Gesamtbild eine Vierersymmetrie (Bild 9, 3. US). Jede Veränderung dieser Anfangswerte bis hin zu punktweisen Neuzuweisungen, wie im abgedruckten Programm, erzeugt andere Ausgangsbilder („Schnittebenen" der Funktion) mit neuartigen Details.

Durch Beseitigen der Kommentarklammern bei Zeilen 53, 59, 60 und Einordnen der Zeilen 39, 40 nach Zeile 46 wird der Doppelstern berechnet.

RISC-Architektur – Eine Übersicht

Prof. Dr. Dieter Jungmann
Informatik-Zentrum an der Technischen Universität Dresden, Wissenschaftsbereich Rechnersysteme

Nachdem in der Mikroprozessortechnik Heft 8/1987 die Frage „Was ist eine RISC-Architektur?" kurz beantwortet wurde, soll mit diesem Beitrag eine Übersicht geboten werden, die Ansatzpunkte für weiterführende Studien bietet.

Das Thema *RISC-Architektur* ist seit Anfang der 80er Jahre in den Gebieten *Rechnerarchitektur, Mikroprogrammierung* und *Compilerimplementierung* eines der häufigsten Themen.

Zentrales Problem ist dabei, ob die Computer mit komplexem Befehlssatz (CISC – Complex Instruction Set Computer), z. B. Computer-Modelle der Architektur IBM/370, VAX11 bzw. Mikroprozessoren der Architektur Motorola 68000 und Intel iAPX 386, durch Computer bzw. Mikroprozessoren mit reduziertem Befehlssatz (RISC = Reduced Instruction Set Computer) verdrängt werden. In Konferenzen und Fachzeitschriften, insbesondere der *Computer Architecture News*, werden neue Ergebnisse der RISC-Architektur-Forschung, -Entwicklung und -Anwendung vorgestellt sowie in Relation zur CISC-Architektur bzw. CISC-Architektur-Implementierung diskutiert. Dieser Beitrag versucht, die Hauptmerkmale der RISC-Architektur in Relation zur CISC-Architektur zu erläutern, einige typische RISC-Implementierungen kurz vorzustellen und eine Einschätzung der perspektivischen Entwicklung zu geben.

Einordnung der RISC-Architektur

In den 70er Jahren hatte sich international die Anwendung höherer Programmiersprachen (HLL = high-level-language) generell durchgesetzt, und nur in Ausnahmefällen, wie dem Kern eines Betriebssystems, wurde noch die Assemblersprache angewendet. *Wirth* verzichtete in seinem Personalcomputer-Projekt LILITH /1/ auf der Basis der Systemprogrammiersprache Modula-2 generell auf die Assemblersprache.

Mit der allgemeinen Zielstellung: *Reduzierung der semantischen Lücke zwischen HLL und ISP* (Instruction Set Processor) *bzw. der Hardware* entstand innerhalb der Rechnerarchitektur das Teilgebiet *HLL-Architektur* mit einer Vielzahl von Architekturimplementierungen, den HLL-Maschinen.

Bis zur Veröffentlichung von *Patterson /2/* galt als generell akzeptiert, daß das ISP-Niveau durch Hardware, insbesondere aber Firmware (Mikroprogramme) auf das Niveau von HLL-Anweisungen angehoben wird. Im Ergebnis umfangreicher statistischer Untersuchungen, wie in /3/, wird nachgewiesen, daß gerade das Gegenteil, ein auf schnelle und leistungsfähige Befehle reduzierter Befehlssatz, die ideale HLL-Architektur ist.

Typische Merkmale

In Übereinstimmung mit umfangreichen Abhandlungen zum Thema RISC-Architektur (z. B. in /4/ und /5/) sind folgende Merkmale in ihrer Einheit bestimmend:

① **Ein-Zyklus-Operationen**
Es werden bis auf die Ausnahme ② nur solche Befehle ausgewählt, die in einem Zyklus ausführbar sind; damit werden schnelle Ausführung und zusätzlich effektive Pipeline-Organisation ermöglicht.

② **Load/Store-Architektur**
Es ist nur der Transport zwischen einem oder mehreren Registersätzen und dem Speicher zulässig; es sind nur 2 Maschinentakte nötig, aber kein orthogonales Befehlssystem.

③ **Verzicht auf Microcode**
Die Merkmale ① und ② ermöglichen und erfordern zugleich, daß anstelle der bisher üblichen Mikroprogramme eine festverdrahtete Steuerung verwendet wird.

④ **Wenige Befehle und Adressierungsarten**
Dadurch wird eine kompakte Codierung und schnelle Ausführung möglich. Typische Befehle sind:
ADD, SUB, AND, OR, XOR, SLL, SRL, SRA, LOAD, STORE, CALL, RET, IMP.
Typische Adressierungsarten sind: Register, Register-Indirekt, Direktwert, PC-relativ, Indexmodifikation.

⑤ **Einheitliches Befehlsformat**
Dadurch wird ebenfalls eine effektive Dekodierung möglich, verursacht wird aber ein erhöhter Speicherplatzaufwand (generell 32 Bit pro Befehl ist typisch).

⑥ **Aufwandsverlagerung in den Compiler**
Dazu zählt die Überwachung und Optimierung der Pipelineorganisation und die Softwareimplementierung komplexer Datentypen.

⑦ **Reduzierung komplexer zugunsten regulärer Hardware-Strukturen**
Dieses Merkmal ist insbesondere für RISC-Mikroprozessoren als VLSI-Realisierung von Bedeutung: drastische Reduzierung des Entwurfsaufwandes, Erhöhung der Ausbeute, maximale Verarbeitungsleistung pro Chipfläche. *Li* bezeichnet in /6/ den RISC-Entwurf aufgrund des geringen Anteiles an irregulärer Steuerlogik und eines hohen Anteiles an regulärer Speicherlogik durch große und mehrere Registersätze als *Data-Storage-(DS-)* Methode. Die Anwendung mehrerer Registersätze ist jedoch kein typisches RISC-Merkmal. Beim gegenwärtigen Entwicklungsstand wird bei RISC jedoch die Implementierung begünstigt.

Implementierungen

Obwohl es allgemein anerkannte RISC-Merkmale gibt, hat sich keine einheitliche RISC-Architektur auf ISP-Ebene durchgesetzt. Damit wird die bestehende Vielfalt an 32-Bit-Mikroprozessoren /7/, /8/ noch erweitert. Nach bisherigen Erfahrungen aus der 8- und 16-Bit-Mikroprozessortechnik setzen sich jedoch nur wenige Mikroprozessorarchitekturen durch. Nach *Tredennick* /9/ werden es im Höchstfall 5 verschiedene 32-Bit-Architekturen sein. Nachfolgend werden Kurzcharakteristiken einiger RISC-Projekte dargestellt.

IBM 801
- Erste experimentelle RISC-Kleinrechner-Implementierung auf der Basis von ECL-Standardschaltkreisen
- Beginn der Arbeiten bereits 1975
- industrielle Überführung in der Workstation IBM 6150, die in den USA als IBM PC/RT (RT = RISC-Technologie) bezeichnet wird
- Verwendung als VLSI-Prozessorelement in einem Parallelrechner-Forschungsprojekt

RISC I/II
- Erste VLSI-Implementierung von 1980-1982 an der Universität Berkeley /3/
- zielt auf effektive C-Implementierung sowie UNIX als Betriebssystem
- Ziele von Folgeprojekten:
  – RISC II hat Cache und Befehlsexpander für häufige 16-Bit-Befehle
  – Smalltalk-HLL-Maschine mit SOAR-Prozessor – Prototyp des von Sun entworfenen und von Fujitsu produzierten SPARC
  – SPUR-Projekt: Mehrprozessorsystem für Symbolverarbeitung

MIPS
- Zunächst Forschungsarbeiten am MIPS-VLSI-Prozessor von Stanfort
- Besonderes Merkmal ist die Pipeline-Überwachung durch den Compiler
- Die Entwickler gründeten die Firma MIPS.
- Die industrielle Entwicklung hat 10 MIPS und ist als Spezialprozessor für OEM-Einsatz vorgesehen (MIPS = Million Operationen pro Sekunde).

Harris Corporation/HSX-7
- In /10/ wird ein RISC-Superminicomputer auf der Basis des STTL-Bit-Slice-Systems AM2900 vorgestellt, das über eine Leistung von 7 MIPS verfügt.

Ridge Computers/Bull SPS 9
- Kleinrechnerimplementierung auf der Basis von High-Speed-Schottky-TTL (400 LSI/MSI-Schaltkreise)

Pyramid Technology Corp./Nixdorf-Targon 35
- Mehrbenutzersystem auf UNIX-Basis mit STTL
- Leistung 3,2 MIPS
- Zwei-Prozessor-Modell vorgesehen

HP-Precision Architecture
- Es wurde ein Familienkonzept auf der Basis einer Architektur und mehrerer Schaltkreistechnologien geplant (von CMOS über ECL bis GaAs)
- z. Z. max. 6,7 MIPS
- Die Kompatibilität zur Vorgänger-CISC-Architektur wird durch zeituneffektive Simulation gewährleistet.

Inmos-Transputer T414
- Ein-Chip-Mikrorechner mit 2 KByte RAM und 4 Link-Kanälen zur Realisierung von hochgradigen Parallelrechnern
- von Floating-Point-Systems auf der Basis der Hypercube-Technologie in Supercomputern im *GFlops*-Bereich eingesetzt (GFLOPS = Milliarde Gleitkomma-Operationen pro Sekunde).

AMD 29000/29300
- CMOS- bzw. bipolare Implementierung von *Function-Slice-Elementen*
- Es sind sowohl RISC- als auch CISC-Prozessoren realisierbar.

Fairchild/Clipper
- *Clipper* /11/ ist ein Hochleistungs-CMOS-Mikroprozessor mit einer Taktfrequenz von 33 MHz
- Der Spitzenleistung von 33 MIPS steht eine mittlere Leistung von 5 MIPS gegenüber
- wird ebenfalls unter UNIX betrieben.

Bewertung und Trendeinschätzung

Wilkes, der Urheber der Mikroprogrammierung und damit der CISC-Prozessoren, schätzt in /12/ ein, daß die Codeverschwendung bei RISC auf längere Sicht nicht ver-

nachlässigbar ist. Tredennick, der Hauptentwickler des MC 68000 und der Micro 370 von IBM, schätzt in /9/ ein, daß RISC-VLSI-Prozessoren lediglich einen festen Platz bei der konsequenten Fortsetzung der Bit-Slice-Mikroprogrammierungs-Kultur haben (AM29000/AM29300). Die Hauptlinien der Entwicklung werden Erweiterungen gegenwärtig dominierender CISC-Mikroprozessoren, wie MC 68030 und japanische 32-Bit-Mikroprozessoren, sein.
Diese Einschätzung wird von Tredennick auf der Basis von Silizium-Halbleitertechnologien getroffen. Erste Forschungsarbeiten auf dem Gebiet der GaAs-Prozessoren haben zunächst eindeutig der RISC-Architektur wegen des geringeren Integrationsniveaus und des großen Anstieges der Laufzeiten in Abhängigkeit vom Signalweg den Vorrang gegeben.
Universelle RISC-Prozessoren werden eindeutig von der Klasse der UNIX-kompatiblen Betriebssysteme unterstützt und sind damit mindestens von der weiteren Verbreitung der reinen UNIX-Anwendung abhängig. Zu beachten ist dabei jedoch, daß der zur Zeit weit verbreitete 32-Bit-Mikroprozessor MC 68020 ebenfalls überwiegend mit UNIX betrieben wird.
Die Anwendung von speziellen RISC-Prozessoren, wie dem Transputer, zur Realisierung hochgradiger Multiprozessorsysteme in der Klasse Supercomputer hat Aussicht auf einen dauerhaften Erfolg, weil damit die Möglichkeit besteht, mit steigendem Integrationsgrad eine maximale Anzahl von Prozessoren auf einem Chip zu integrieren.
Der Marktführer für 32-Bit-Mini-, Supermini- und Mehrplatzsysteme DEC gibt in diesem Sinne ebenfalls zu verstehen, daß die RISC-Architektur keine Alternative zur VAX-Architektur darstellt.
Die Hauptarchitekturlinie Japans kann /13/ und /14/ zufolge TRON heißen. Die diesem Projekt zugrundeliegende TRON-VLSI-CPU hat eine eindeutige CISC-Charakteristik, ermöglicht aber die Abarbeitung der am häufigsten verwendeten CISC-Untermenge nach einer RISC-Charakteristik. Der Architekturentwurf ist als offenes Konzept analog OSI langfristig angelegt, denn es ist nach den ersten 32-Bit-VLSI-Implementierungen bereits ein Ausbau zu 48-Bit-Adressierungsweite und einer vollen 64-Bit-Architektur vorgesehen.
Das in Japan erreichte technologische Niveau in der Mikroelektronik- und Computerproduktion sowie die seit 1986 übernommene Führungsrolle in der Mikroelektronikproduktion in Verbindung mit dem Architekturkonzept TRON kann auch zur Beherrschung des Computermarktes führen, insbesondere in den Positionen Hochleistungscomputer und Industriecomputer.

Zusammenfassung

Prozessoren und Systeme mit RISC-Architektur wurden in relativ kurzer Frist und großer Breite aus dem Forschungsstadium in industrielle Entwicklungen überführt. Besondere Bedeutung hat die RISC-Architektur für 32-Bit-VLSI-Prozessoren in Verbindung mit dem Betriebssystem UNIX in der Computer-Leistungsklasse Workstation erlangt. Trotz der hohen Leistungsfähigkeit, der einfachen Entwerfbarkeit und hohen Ausbeute durch weitgehend reguläre Strukturen haben Mikroprozessoren mit 32-Bit-RISC-Architektur im Vergleich zu bereits länger eingeführten Mikroprozessoren mit CISC-Architektur, wie MC 68000/68020, iAPX 286/386 und VAX11/MicroVAXII, keinen Durchbruch erzielt. Mit zunehmendem Integrationsgrad und dem Erscheinen von völlig neuen CISC-Architekturen mit RISC-Anteil, wie die auf 64 Bit vorbereitete 32-Bit-TRON-Architektur der Japaner, sinkt die Wahrscheinlichkeit für eine RISC-Dominanz. Dagegen werden RISC-Spezialprozessoren, wie der Transputer, eine langfristige Perspektive haben.
Bei Hochgeschwindigkeitstechnologien wie ECL und GaAs ermöglicht die RISC-Methode, zu einem frühen Zeitpunkt 32-Bit-Einchip-Prozessoren zu realisieren oder bei der CMOS-Technologie die mit einem geringeren Entwicklungsaufwand verbundene Gate-Array-Entwurfsmethode anzuwenden.

Literatur

/1/ Wirth, N.: LILITH: A Personal Computer for the Software Engineer. Microcomputer System Design, LNCS 126, Springer-Verlag 1982
/2/ Patterson, D. A.; Ditzel, D. R.: Retrospective on High-Level Language Computer Architecture. 7th Annual Symp. on Comp. Architecture 1980
/3/ Patterson, D. A.; Piepho, R. S.: RISC-Accessment: A High-Level-Language Experiment. 9th Int. Symp. on Comp. Arch. 1982
/4/ Calwell, R. D.: Instruction Sets and Beyond: Computers, Complexity and Contraversy. Computer-Journal Sept. 1985, S. 8
/5/ Klein, A.: Redused Instruction Set Computers – Grundprinzipien einer neuen Prozessorarchitektur. Informatik-Spektrum, Band 9, Heft 6, Dez. 1986
/6/ Li, T.: A VLSI View of Microprogrammed System Design. 15th An. Micropr. Workshop (MICRO 15) 1982, S. 96
/7/ Myers, G. J.; Yu, Y. C.; Hanse, D. L.: Microprozessor Technology Trends. Proc. of the IEEE, Vol. 74 (1986) 12, S. 1605
/8/ Milutinovic, F.; Silbey, G.: A Survey of Advanced Microprocessors and HLL-Computer Architectures. Computer, Aug. 1986, S. 72
/9/ Tredennick, N.: The Impact of VLSI on Microprogramming. MICRO (1986) 19, S. 2
/10/ Killman, P.: Minicomputer Takes Simple Route to Speed UNIX. Computer Design 33 (1985) 24
/11/ Sachs, H.: A High Performance 846000 Transistor UNIX Engine – The Fairchild Clipper. Proceedings ICCD85 1985
/12/ Wilkes, M.: The Processor Instruction Set. MICRO (1982) 15
/13/ Sakamura, K.: The TRON Project. IEEE Micro, April 1987, S. 8
/14/ Sakamura, K.: Architecture of the TRON VLSI CPU. IEEE Micro, April 1987, S. 17

✉ KONTAKT ☏

Informatik-Zentrum des Hochschulwesens an der TU Dresden, WB Rechnersystem/LB Rechnerarchitektur, Mommsenstraße 13, Dresden, 8027; Tel. 4575321

Filterprogramme und Pipes unter MS-DOS

Frank Lindner, Weimar

Der Artikel behandelt die grundlegenden Voraussetzungen des Betriebssystems MS-DOS zur E/A-Umlenkung und die daraus resultierenden Möglichkeiten der Nutzung von Filtern und Pipes.
Insbesondere wird die Erstellung von Filterprogrammen mit TURBO-PASCAL (V.3.01) dargestellt und an einem Beispiel erläutert. Als Schlußteil folgt eine Darstellung der Mechanismen auf Assemblerebene, ebenfalls mit einem Beispiel.

Grundlagen

Voraussetzung für die Funktion von **Filtern** und **Pipes** ist die Möglichkeit der Ein-/Ausgabe-Umlenkung (E/A-Umlenkung). Der Vorgang der E/A-Umlenkung wird als I/O-Redirection bezeichnet. Sein Ursprung liegt im Betriebssystem UNIX /1/. Er ist Konsequenz der Realisierung moderner Systemkonzepte, wie Modularisierung und Hardwareunabhängigkeit.
Den Ausgangspunkt bildet die Definition logischer Ein-/Ausgabe-Kanäle. Über diese Kanäle realisiert der Problemprogrammierer die Ein- und Ausgabe seiner Daten. Damit wird der Programmierer unabhängig von den physischen Quellen und Zielen. Für die Zuordnung dieser standardisierten Kanäle zu den entsprechenden Geräten sorgt dann die Systemsoftware der konkret konfigurierten Anlage des Anwenders. Die vorgegebenen Kanäle sind:
Standardeingabe, -ausgabe, -fehlerausgabe, und -listgerät sowie standard-auxiliary I/O.
Gibt also ein Programm seine Fehlermeldungen über die Standardfehlerausgabe aus, kann es davon ausgehen, daß diese Meldungen auf einem Gerät abgebildet werden, auf welchem die Fehlermeldung wahrgenommen wird. Ob dieses Gerät in der genutzten Anlage ein Bildschirm, Drucker o. a. ist, ist für die Programmierung ohne Belang.
Während die Zuordnung der Kanäle Standardfehlerausgabe, -listgerät und standard-auxiliary I/O vom Nutzer der Anlage nicht geändert werden kann, gibt es für die Standardeingabe und -ausgabe, die beide üblicherweise dem Bedienerterminal zugeordnet sind, die Möglichkeit der Zuordnungsänderung. Diese Möglichkeit ist der Inhalt der E/A-Umlenkung.
Die Umlenkung erfolgt durch die Angabe folgender Operatoren in der Eingabezeile des Kommandoprozessors (COMMAND.COM):
< infile – Eingabe erfolgt von Datei infile
> outfile – Ausgabe erfolgt auf Datei outfile; outfile wird neu angelegt
>> outfile – Ausgabe wird an bestehende Datei outfile angehängt
pl|p2|... – Ausgabe von Programm pl wird als Eingabe an Programm p2 weitergegeben (Pipeoperator) und so fort...
Bemerkung: Als infile bzw. outfile können auch unter MSDOS gültige Gerätenamen verwendet werden, sofern dies sinnvoll ist.
Beispiele:
type ascii.doc > PRN
type ascii. mst > NUL
Aus der Verwendung der angegebenen Zeichen als derartige Operatoren erklärt sich das Verbot ihrer Nutzung in Dateinamen. Die Anweisungen zur E/A-Umlenkung können an beliebiger Stelle in der Kommandozeile aufgeführt werden.
Systeminterne Grundlage der Realisierung dieser Konzeption bildet die gleichartige Behandlung von Dateien und E/A-Kanälen an der Programmierschnittstelle des Betriebssystems.
Die Systemfunktionen von MS-DOS, ab Version 2.xx, lassen sich einteilen in die Gruppe der CP/M-ähnlichen Systemfunktionen und

die Menge der XENIX-ähnlichen Systemfunktionen (XENIX ist ein UNIX-kompatibles Betriebssystem der Firma Microsoft, von der ebenfalls MS-DOS vertrieben wird).
Das Vorhandensein dieser beiden Gruppen von Systemfunktionen ist historisch begründet (siehe dazu /2/).
Durch die Aufnahme der XENIX-ähnlichen Funktionen ab Version 2.xx in den Leistungsumfang von MS-DOS wurden die oben genannten notwendigen Grundlagen geschaffen.
Als Filterprogramme werden Softwarelösungen verstanden, die Daten von der Standardeingabe empfangen, in einer beliebigen Art und Weise manipulieren und auf die Standardausgabe ausgeben.
Diese Programme lassen sich miteinander zu sogenannten Pipes (Röhren) in der Art verketten, daß jeweils die Ausgabe auf den *stdout*-Kanal des Vorläuferprogramms die Eingabe über den *stdin*-Kanal für das folgende Programm darstellt. Die Daten werden sinngemäß durch eine *Röhre* gesandt und während des Durchlaufs in den verschiedenen *eingebauten Filtern* manipuliert.
Beispiele für Filterprogramme sind die zum Systemumfang zählenden externen MS-DOS-Befehle SORT, FIND und MORE.
Bei der Abarbeitung von Pipes werden von MSDOS Zwischendateien zur Pufferung der Daten auf externen Speichermedien angelegt.
Unter Version 2.xx erhalten diese Zwischendateien Dateinamen der Form %PIPExxx.$$$, in den Versionen 3.xx wird wegen der Netzwerkfähigkeit und der damit möglichen Zugriffskonflikte der Name aus der aktuellen Systemzeit gebildet (Stunden, Minuten, Sekunden, Hundertstelsekunden in hexadezimaler Form).
Diese Dateien werden von MS-DOS selbst wieder entfernt.
Das Anfertigen von Filterprogrammen und ihre Kombinationsmöglichkeit stellt eine geeignete und schnelle Methode dar, Anforderungen des Nutzers bzw. von Nutzerprogrammen gerecht zu werden, ohne bestehende Lösungen komplex bearbeiten zu müssen.
Darüber hinaus ist es möglich, vorhandene Lösungen für eigene Zwecke zu nutzen und Programme zu erstellen, die oft und flexibel genutzt werden können. Damit wird die Modularisierung von einzelnen Programmfunktionen in kleinen Programmeinheiten unterstützt.
Ein deutliches Beispiel hierfür ist der Filter MORE, der die seitenweise Datenausgabe, bezogen auf den Bildschirm, für den Standardausgabekanal organisiert.
DIR|MORE bewirkt die seitenweise Ausgabe des Inhaltes der Diskette;
Type text|MORE die seitenweise Ausgabe des Inhaltes der Datei **text**.
Einige Bemerkungen zur Programmierung sollen im folgenden gemacht werden. Zu Beginn stehen Aussagen, die speziell die Programmierung von Filterprogrammen in **TURBO-PASCAL** erläutern.
Um den Mechanismus allgemeingültiger darstellen zu können, im Anschluß einige Erklärungen für die systemnahe Programmierung und ein in Assembler gefertigtes Beispiel.
Zusätzlich sind Informationen zu den notwendigen Systemfunktionen gegeben, um bei Bedarf Zusätze für andere Sprachen erstellen zu können oder auch um auf Assemblerebene programmieren zu können.

Bild 1 Nutzung der Standardkanäle mit Turbo-Pascal (V. 3.xx)

```
program StdIO;

type Wrkstring = String[80];

var  Std_INP,
     Std_OUT,
     Std_ERR    : Text;
     S          : Wrkstring;

begin
     assign   ( Std_INP,'INP:' )  ;{ Filevariablen zuweisen }
     assign   ( Std_OUT,'OUT:' )  ;
     assign   ( Std_ERR,'ERR:' )  ;
     reset    ( Std_INP )  ;          { Oeffnen der Kanaele }
     rewrite  ( Std_OUT )  ;
     rewrite  ( Std_ERR )  ;
     readln   ( Std_INP,S )  ;        { Benutzung ... }
     writeln  ( Std_OUT,S )  ;
     writeln  ( Std_ERR,S )  ;
     close    ( Std_INP )  ;          { Schlieszen der Kanaele }
     close    ( Std_OUT )  ;
     close    ( Std_ERR )  ;
end.   { StdIO }
```

```
 1: {___________________________________________________________________
 2:
 3:  FLI                 TABFIL  ( VERS. 2.0 )             11. Oktober 1987
 4:  ___________________________________________________________________
 5:
 6:  PROGRAMMERSTELLER:  F. Lindner    5300 Weimar  Th.- Muentzer - Str.7
 7:
 8:  LEISTUNG:           Dieser Filter kopiert von der Standardeingabe nach
 9:                      der Standardausgabe und formt dabei alle Tabs in
10:                      Leerzeichen um.
11:                      Aufruf :  TABFIL < Alt.Ext > Neu.Ext
12:  ___________________________________________________________________}
13:
14: {$G16384,P16384,D-}       { Direktiven setzen, Puffer jeweils = 16kByte }
15:
16: program TabFil;
17:
18: var       S                          : String[255];
19:           i                          : Byte;
20:           InS                        : String[8];
21:           Pos_Tab, Spc,
22:           Anf_Such, Ber_Pos, C_Chr : Integer;
23:
24: const     Tab = ^I;
25:
26: begin
27:      while not EoF do begin                      { bis Ende des Files ..... }
28:        ReadLn ( S );                             { zeilenweise Lesen ...... }
29:        Anf_Such := 1;                            { Laufvar. initialisieren. }
30:        C_Chr    := 0;                            { bis jetzt keine Steuerz. }
31:        Pos_Tab  := Pos ( Tab, S );               { ersten Tab suchen ...... }
32:        while Pos_Tab <> 0 do begin               { bis keine Tab's mehr in S}
33:          for i:= Anf_Such to Pos_Tab - 1 do      { Steuerzeichen suchen ... }
34:          begin
35:            case S [i] of
36:              ^A..^G : C_Chr := C_Chr + 1;        { allgemeine Steuerzeichen }
37:                  ^H : C_Chr := C_Chr + 2;        { BACKSPACE .............. }
38:              ^J..^_ : C_Chr := C_Chr + 1;        { allgemeine Steuerzeichen }
39:            end;   { case ... }
40:          end;   { for i... }
41:          Ber_Pos := Pos_Tab - C_Chr;             { Steuerz. beruecksichtigen}
42:          Spc := ( 8 - ( Ber_Pos mod 8 ) ) and 7; { Anz. Leerz. ermitteln}
43:          InS := '';                              { String z. Einf. loeschen }
44:          for i := 1 to Spc do InS := InS + ' '; { String z. Einf. aufb. }
45:          S [ Pos_Tab ] := ' ';                   { Tab loeschen .......... }
46:          Insert ( InS, S, Pos_Tab );             { Leerzeichen einfuegen .. }
47:          Anf_Such := Pos_Tab + Spc + 1;          { Durchsuchen zwecklos ... }
48:          Pos_Tab := Pos ( Tab, S );              { naechsten Tab suchen ... }
49:        end;   { while Pos_Tab... }
50:       WriteLn(S);                                { zeilenweise Ausgeben ... }
51:      end;   { while not EoF... }
52: end.  { TabFil }
```

Bild 2 TabFil – ein Filter in Turbo-Pascal (V. 3.xx)

Filter in TURBO-PASCAL

TURBO-PASCAL (Version 3.xx) unterstützt in zwei Formen die oben erläuterten Mechanismen.
In der Implementation des TURBO-PASCAL System für MS-DOS werden die Möglichkeiten der E/A Umlenkung gezielt unterstützt. Neben dieser Unterstützung existieren drei dem TURBO-PASCAL-System bekannte logische Geräte (INP:, OUT:, ERR:), die den expliziten Zugriff auf die Standardeingabe, -ausgabe und -fehlerausgabe ermöglichen. Die Zuordnung der logischen Geräte zu den genannten E/A-Kanälen ist der Mnemomik eindeutig zu entnehmen. Allerdings existieren für diese Geräte keine Standardfilevariablen, wie AUX für COM1:. Die Filevariablen müssen vom Programmierer daher selbst zugewiesen, die Files eröffnet und geschlossen werden. Die genannten Geräte können beliebigen typisierten und nichttypisierten Files zugeordnet werden. Das grundsätzliche Vorgehen zeigt Bild 1.
Die Programmiermethode zur Nutzung der E/A Umlenkung basiert auf drei Compilerdirektiven.
$D (Voreinstellung: $D+)
Diese Direktive beeinflußt die Art der Ein- und Ausgabe auf folgende Weise:
Ist die Direktive aktiv ($D+), erfolgt beim Eröffnen einer Textdatei (Reset, Rewrite, Append) eine Anfrage an das Betriebssystem,

ob es sich um eine Plattendatei oder ein Gerät handelt.
In Abhängigkeit vom Resultat erfolgt die E/A von und zu Geräten zeichenweise bzw. von und zu Dateien gepuffert. Wird diese Compilerdirektive negiert ($D-), erfolgt die Anfrage nicht, und eine gepufferte Ein- und Ausgabe auf logischen Geräten zugeordneten Dateien wird möglich.
Mit dem Aufruf der Flush-Prozedur kann in diesem Fall die restlose Ausgabe des Pufferinhaltes gesichtert werden.
$Gx (Voreinstellung: $G0; G = Get)
Diese Direktive dient dem Anlegen eines Eingabepuffers; x ist ein Platzhalter für die geforderte Größe des Puffers in Byte. Ist die Puffergröße 0, wird die Eingabedatei (INPUT) CON: oder TRM: zugeordnet. Die Standardeingabe von MS-DOS wird benutzt, wenn die Puffergröße größer 0 ist.
$Px (Voreinstellung: $P0; P = Put)
Bei dieser Compilerdirektive steht x für die Größe des Ausgabepuffers in Byte. Ist die Puffergröße 0, wird die Ausgabedatei (OUTPUT) CON: oder TRM: zugeordnet. Die Standardausgabe von MS-DOS wird benutzt, wenn die Puffergröße größer 0 angegeben wird.
Alle drei Compilerdirektiven müssen am Anfang des Programms, noch vor dem Deklarationsteil, stehen und können demgemäß innerhalb eines Programms nicht geändert werden!
Bild 2 zeigt das Listing eines Filters als TURBO-PASCAL-Quelltext (TABFIL).
Aus diesem Beispiel kann der an sich unproblematische *Rahmen* eines Filterprogramms entnommen werden. Er besteht aus erstens dem Setzen der Compilerdirektiven in Zeile 14 und zweitens der „while not EoF"-Schleife in Zeile 27.
Funktion dieses Filters ist es, die in einem Textfile enthaltenen Steuerzeichen für die Ausführung von Horizontaltabulatorsprüngen (ASCII-Code=09) durch die notwendige Anzahl von Leerzeichen zu ersetzen.
Dies ist insbesondere dann sinnvoll, wenn man Textfiles, die das Steuerzeichen 09 enthalten mit dem TURBO-Editor bearbeiten möchte. Der Tabulatorsprung wird von diesem Editor nicht ausgeführt, sondern als Steuerzeichen ^I angezeigt. Besonders deutliche Beispiele sind .LST-Files, die in jeder Zeile mehrere Tabulatoren enthalten. Zusätzlich berücksichtigt wird die Wirkung des Steuerzeichens 08 (Backspace: löscht das vorhergehende Zeichen) und das Vorhandensein von Steuerzeichen, die selbst keine gültigen „Textzeichen" darstellen.
Grundsätzlich ist auch die E/A für Programme umlenkbar, die Eingabe von CON:, TRM: oder KBD: erwarten bzw. Ausgaben über diese logischen Geräte realisieren. Für die Eingaben wird dann die Funktion ConIn genutzt, die ihrerseits in MSDOS-Funktion 8 aufruft und für die Ausgaben die Prozedur ConOut, welche die MSDOS-Funktion 2 verwendet.
Beides sind *CP/M-ähnliche* MSDOS-Systemfunktionen.
Da ein Filter die Programmbeendigung in aller Regel vom Ende der bearbeiteten Datei abhängig machen muß, kommt der Funktion EoF tragende Bedeutung zu. Bei Verwendung dieser Funktion in Zusammenhang mit den logischen Geräten CON:, TRM: und KBD: gibt EoF jedoch nur dann den Wert **wahr** zurück, wenn das Dateiendekennzeichen ^Z (=1AH) erkannt wird.

```
  1: PAGE      65,120
  2: TITLE     UPCASE - Umwandlung von Klein- in Groszbuchstaben
  3: NAME      UPCASE
  4: ;         AUTOR:              F. Lindner       5300 Weimar   Th.-Muentzer-Str. 7
  5: ;         ASSEMBLER:          MASM    V. 4.0  (LINK    V. 3.04)
  6:
  7: HANDLE_IN          EQU        00H      ;Filehandle fuer Standardeingabe
  8: HANDLE_OUT         EQU        01H      ;Filehandle fuer Standardausgabe
  9: HANDLE_ERR         EQU        02H      ;Filehandle fuer Standardfehlerausgabe
 10: MSDOS              EQU        021H     ;Interrupt fuer verwendete MSDOS-Funkt.
 11: READ               EQU        03FH     ;Funktionsnummer fuer Lesen
 12: WRITE              EQU        040H     ;Funktionsnummer fuer Schreiben
 13: EXIT               EQU        04CH     ;Funktionsnummer fuer Programmbeendigung
 14: REQ_NUM            EQU        4096     ;angeforderte Zeichenzahl = Puffergroesze
 15: a                  EQU        061H     ;Code fuer 'a' im 7-Bit ASCII-Zeichensatz
 16: z                  EQU        07AH     ;Code fuer 'z' im 7-Bit ASCII-Zeichensatz
 17: CR_LF              EQU        0A0DH    ;Code fuer Carrige Return & Line Feed
 18:
 19: ;Datenbereich...
 20:
 21: DATA     SEGMENT 'DATA'
 22:
 23: ERR_TXT1           DW         CR_LF,
 24:                    DB         '*** UPCASE: Eingabefunktion (3FH) meldet Fehler'
 25:                    DB         ' beim Einlesen von der Standardeingabe',
 26:                    DW         CR_LF
 27: LEN_TXT1 = $ - OFFSET ERR_TXT1   ;Textlaenge ermitteln
 28: ERR_TXT2           DW         CR_LF,
 29:                    DB         '*** UPCASE: Ausgabefunktion (40H) meldet Fehler'
 30:                    DB         ' beim Schreiben auf die Standardausgabe',
 31:                    DW         CR_LF
 32: LEN_TXT2 = $ - OFFSET ERR_TXT2
 33: ERR_TXT3           DW         CR_LF,
 34:                    DB         '*** UPCASE: allgemeiner Fehler beim Schreiben',
 35:                    DW         CR_LF,
 36:                    DB         '*** --""--: Disk voll ?'
 37:                    DW         CR_LF,
 38:                    DB         '*** --""--: Ausgabe auf Geraet (PRN:/CON:..) ?',
 39:                    DW         CR_LF
 40: LEN_TXT3 = $ - OFFSET ERR_TXT3
 41: BUFFER             DB         REQ_NUM DUP (?)         ;Puffer fuer E/A
 42: ERR_LEVEL          DB         0       ;Beendigungscode fuer Umgebung
 43: ACT_NUM            DW         0       ;aktuell gelesene Zeichenzahl
 44: EOF                DB         0       ;Kennzeichen fuer End of File
 45:
 46: DATA      ENDS
 47:
 48: ;Programmcode...
 49:
 50: CODE     SEGMENT 'CODE'
 51: ASSUME   CS:CODE, DS:DATA
 52: BEGIN:
 53:              MOV      AX,DATA
 54:              MOV      DS,AX
 55:              MOV      ES,AX                ;Daten- und Extrasegment initialisieren
 56:              CALL     STD_IN               ;Zeichen in Puffer lesen
 57:              CMP      ERR_LEVEL,00         ;Fehler aufgetreten ?
 58:              JNE      EXIT_PROG            ;wenn ja Programm abbrechen
 59:              MOV      CX,ACT_NUM           ;Anzahl der gelesenen Zeichen nach CX
 60:              CMP      CX,00                ;kein Zeichen gelesen ?
 61:              JE       EXIT_PROG            ;Programm beenden
 62:              CMP      CX,REQ_NUM           ;volle geforderte Zeichenanzahl gelesen ?
 63:              JE       CONTINUE
 64:              MOV      EOF,0FFH             ;wenn nicht --> Endekennzeichen laden
 65: CONTINUE:
 66:              MOV      DI,OFFSET BUFFER         ;Beginn Puffer nach DI
 67: LOAD:
 68:              MOV      AL,[DI]              ;Zeichen holen
 69:              CMP      AL,a                 ;Codes vergleichen: < a ?
 70:              JB       NO_CONV              ;wenn ja keine Konvertierung notwendig
 71:              CMP      AL,z                 ;Codes vergleichen: > z ?
 72:              JA       NO_CONV              ;wenn ja keine Konvertierung notwendig
 73:              SUB      AL,020H              ;konvertieren
 74:              MOV      [DI],AL              ;und zurueckspeichern
 75: NO_CONV:
 76:              INC      DI                   ;Pufferzeiger weitersetzen
 77:              DEC      CX                   ;Zaehler := Zaehler - 1
 78:              CMP      CX,00                ;alle Zeichen bearbeitet ?
 79:              JNZ      LOAD                 ;wenn nicht, naechstes Zeichen laden
 80:              CALL     STD_OUT              ;Puffer nach Bearbeitung ausgeben
 81:              CMP      ERR_LEVEL,00         ;Fehler aufgetreten
 82:              JNE      EXIT_PROG            ;wenn ja Programm abbrechen
 83:              CMP      EOF,0FFH             ;Dateiende erreicht ?
 84:              JNE      BEGIN                ;wenn nicht, naechsten Block laden
 85: EXIT_PROG:
 86:              MOV      AL.ERR_LEVEL         ;ERRORLEVEL (fuer Umgebung ) setzen
 87:              MOV      AH,EXIT
 88:              INT      MSDOS                ;Programm beenden
 89:
 90:
 91: STD_IN    PROC
 92:
 93:              MOV      AH,READ              ;Funktionsnummer laden
 94:              MOV      BX,HANDLE_IN         ;Filehandle fuer Standardeing. als Kanal
 95:                                            ;zur Eingabe
 96:              MOV      CX,REQ_NUM           ;Anzahl der geforderten Zeichen
 97:              MOV      DX,OFFSET BUFFER         ;Puffer adressieren
 98:              INT      MSDOS                ;Funktion ausfuehren lassen
 99:              MOV      ACT_NUM,AX           ;tatsaechlich geles. Zeichenz. speichern
100:              JNC      RET_IN               ;falls kein Fehler auftrat zurueck
101:              MOV      ERR_LEVEL,01H        ;ansonsten: ERR_LEVEL setzen
102:              MOV      CX,LEN_TXT1              ;   Laenge Fehlertext nach CX
103:              MOV      DX,OFFSET ERR_TXT1       ;   Adresse Fehlertext nach DX
104:              CALL     STD_ERR                  ;   Fehler ausgeben
```

```
105: RET_IN:
106:          RET                           ;zurueck zum Hauptprogramm
107:
108: STD_IN   ENDP
109:
110:
111: STD_OUT  PROC
112:
113:          MOV     AH,WRITE              ;Funktionsnummer laden
114:          MOV     BX,HANDLE_OUT         ;Filehandle fuer Standardausgabe als
115:                                        ;Ausgabekanal
116:          MOV     CX,ACT_NUM            ;Anzahl der zu schreibenden Zeichen
117:          MOV     DX,OFFSET BUFFER        ;Puffer adressieren
118:          INT     MSDOS                 ;Funktion ausfuehren lassen
119: TEST1:   JNC     TEST2                 ;falls kein Fehler --> zu TEST2
120:          MOV     ERR_LEVEL,02H         ;ansonsten: ERR_LEVEL setzen
121:          MOV     CX,LEN_TXT2             ;   Laenge Fehlertext nach CX
122:          MOV     DX,OFFSET ERR_TXT2      ;   Adresse Fehlertext nach DX
123:          CALL    STD_ERR                 ;   Fehler ausgeben
124: TEST2:
125:          CMP     AX,ACT_NUM            ;tatsaechlich geschr. Zeichenzahl pruefen
126:          JE      RET_OUT               ;wenn wie gefordert, ohne Fehler zurueck
127:          MOV     ERR_LEVEL,03H         ;ansonsten: Fehlercode setzen
128:          MOV     CX,LEN_TXT3             ;   Laenge Fehlertext nach CX
129:          MOV     DX,OFFSET ERR_TXT3      ;   Adresse Fehlertext nach DX
130:          CALL    STD_ERR                 ;   Fehler ausgeben
131: RET_OUT:
132:          RET                           ;zurueck zum Hauptprogramm
133:
134: STD_OUT  ENDP
135:
136:
137: STD_ERR  PROC
138:
139:          MOV     AH,WRITE              ;Funktionsnummer laden
140:          MOV     BX,HANDLE_ERR         ;Filehandle fuer Standardfehlerausg. als
141:                                        ;Ausgabekanal
142:          INT     MSDOS                 ;Funktion ausfuehren lassen
143:          RET                           ;zurueck zur aufrufenden Prozedur
144:
145: STD_ERR  ENDP
146:
147: CODE     ENDS
148:
149: END      BEGIN
```

Bild 3 UPCAE – Beispiel eines Filters in 8086-Assembler

(Fortsetzung von S. 265)

Moderne Betriebssysteme, wie auch MS-DOS, benutzen jedoch nicht in jedem Fall das Dateiendekennzeichen, sondern erkennen das Dateiende im anderen Fall mit Hilfe der vom System verwalteten Dateigröße. Endet eine Datei nicht mit ^Z, wird das Dateiende nicht erkannt, der Filter beendet seine Arbeit nicht und liest nach dem physischen Dateiende von der Tastatur. Diese Methode (Enderkennung über Dateigröße) ist deshalb unbrauchbar.

Mechanismus auf Assemblerniveau

Die im Abschnitt Grundlagen erwähnte Gleichartigkeit des Zugriffs der XENIX-ähnlichen Systemfunktionen auf Dateien und Geräte äußert sich für den Programmierer darin, daß er sowohl zum Lesen einer Datei/einem Gerät, wie auch zum Schreiben auf eine Datei/ein Gerät, jeweils die gleiche Systemfunktion benutzt.

Wie erfolgt nun die Identifikation des gewünschten Kanals?

Im Gegensatz zur Dateiarbeit mit den CP/M-ähnlichen Funktionen, welche, wie von CP/M bekannt, mit File-Control-Blöcken (FCBs) arbeiten, identifizieren die erwähnten Systemfunktionen den gewünschten Kanal mit Hilfe einer ihm zugeordneten Nummer. Diese Nummer ist vom Typ WORD (zwei Byte) und wird als **Filehandle** bezeichnet.

Wird mit der entsprechenden Systemfunktion eine Datei eröffnet bzw. wird bei der öffnenden Funktion auf einen gültigen Gerätenamen verwiesen, so wird als Ergebnis ein Filehandle zurückgegeben, mit dessen Hilfe die Datei/das Gerät bei Zugriffsfunktionen identifiziert werden kann.

Ein weiterer großer Vorteil besteht bei dieser Vorgehensweise für den Programmierer darin, daß er keine Einträge in die FCBs mehr pflegen. Die notwendigen Daten werden vom Betriebssystem selbst in internen Datenstrukturen verwaltet.

Für die Standard-E/A-Kanäle stehen vordefinierte Filehandle zur Verfügung.

Diese sind wie folgt vergeben:

- 00 stdin (Standardeingabe)
- 01 stdout (Standardausgabe)
- 02 stderr (Standardfehlerausgabe)
- 03 stdaux (standard-auxiliary I/O) (AUX/COMI)
- 04 stdlst (Standarddruckerausgabe) (PRN/LPTI).

Das erste Filehandle, welches für einen weiteren vom Programmierer angeforderten Zugriffskanal vergeben wird, ist wegen dieser vordefinierten Zuordnung **05**.

Die vordefinierten Filehandle gestatten den sofortigen Zugriff auf die Kanäle, sie werden vom Betriebssystem geöffnet.

Die zu benutzenden Systemaufrufe sind Funktionen des Interrupts 2IH (MSDOS).

Zum Lesen wird die Funktion 3FH (READ) verwendet. Diese Funktion fordert im Register BX ein Filehandle zur Identifikation der Datei/des Gerätes von der/dem gelesen werden soll

(HANDLE_IN=00H=Standardeingabe),

in CX die Anzahl der Zeichen, die gelesen werden sollen (REQ_NUM) und in DX den Offset des Speicherbereiches bezüglich DS, in dem die gelesenen Zeichen abgelegt werden sollen (BUFFER). Außerdem ist Register AH mit dem Wert 3FH zu laden, um den Konventionen für die Systemaufrufe des Interrupts 2IH zu entsprechen.

Nach Ausführung der Funktion enthält das Register AX die Anzahl der tatsächlich gelesenen Zeichen. Ist die Anzahl der gelesenen Zeichen geringer als die geforderte Anzahl, so ist dies beim Lesen von einer Datei als Erreichen des Dateiendes während des Lesevorganges zu werten (siehe dazu Zeilen 59–64 von Bild 3). Traten während der Ausführung der Funktion Fehler auf, wird dieser Zustand durch das gesetzte Carry-Flag signalisiert (siehe Zeile 100 und ERR_TXTI).

Zur genaueren Analyse der Fehlerursache wird auf /2/ verwiesen.

Zur Ausgabe wird die Funktion 40H (WRITE) benutzt. Adäquat zur Funktion 3FH (READ) fordert sie in Register BX ein Filehandle

(HANDLE_OUT=0IH=Standardausgabe,

HANDLE_ERR=02H=Standardfehlerausgabe),

in Register CX die Anzahl der zu schreibenden Zeichen (ACT_NUM) und in DX den Offset des Speicherbereichs bezüglich DS, der die zu schreibenden Zeichen enthält (BUFFER). Register AH ist mit dem Wert 40H (WRITE) zu laden.

Auch diese Funktion gibt nach Rückkehr in AX die Anzahl der tatsächlich, in diesem Fall, geschriebenen Zeichen an und setzt bei Fehlern während der Ausführung das Carry-Flag (ERR_TXT2).

Zur genaueren Bestimmung des Fehlers, den das gesetzte Carry-Flag signalisiert, wird wiederum auf /2/ verwiesen.

Eine Abweichung der Anzahl der tatsächlich geschriebenen Zeichen von der geforderten Anzahl deutet auf einen allgemeinen Fehler beim Schreiben auf die Datei/das Gerät hin.

Dabei ist Folgendes zu beachten:

Erhält die Funktion 3FH ihre Daten aus einer Datei, so liest sie sämtliche Zeichen einschließlich der Dateiendekennzeichen, sofern vorhanden, ein und gibt auch die entsprechende Zeichenanzahl in AX an das aufrufende Programm. Erfolgt die Ausgabe wieder auf eine Datei, werden alle Zeichen, einschließlich der eingelesenen Dateiendekennzeichen, auch tatsächlich geschrieben. Ist die Standardausgabe jedoch einem Gerät zugeordnet (CON: oder PRN:, andere Geräte stehen dem Autor nicht zur Verfügung), endet die Ausgabe mit dem letzten Zeichen vor dem Dateiendekennzeichen (ERR_TXT3)!

Anmerkung: Die in Klammern angegebenen Begriffe beziehen sich auf die im Assemblerquelltext verwendeten Bezeichnungen.

Der in der Assemblersprache des 8086 verfaßte Filter UPCASE (siehe Bild 3) hat die Aufgabe, Klein- in Großbuchstaben zu konvertieren. Zu diesem Zweck liest er Zeichen von der Standardeingabe, testet, ob es sich um den Code eines Kleinbuchstaben bezüglich der ASCII-Codierung handelt, setzt den Code gegebenenfalls um und übergibt die Zeichen an die Standardausgabe. Treten beim Programmablauf Fehler auf, werden die entsprechenden Fehlermeldungen (ERR_TXTx) über die Standardfehlerausgabe ausgegeben.

Zur einfacheren Orientierung wurden die Prozeduren ihren benutzten Kanälen entsprechend bezeichnet. Das heißt, die Prozedur STD_IN übernimmt das Einlesen von der Standardeingabe, STD_OUT gibt über die Standardausgabe aus, und STD_ERR übergibt die Fehlermeldungen an die Standardfehlerausgabe.

Die Ein- und Ausgaben erfolgen gepuffert. Die Puffergröße sollte möglichst groß gewählt werden, um die Anzahl der Plattenzugriffe bei der Umleitung der Ein-/Ausgabe auf eine Datei zu minimieren.

Literatur

/1/ Claßen, L.; Oefler, U.: Unix und C. VEB Verlag Technik, Berlin 1987

/2/ Smode, D.: Das große MS-DOS-Profi-Arbeitsbuch. Franzis' Verlag, München 1987

/3/ Handbuch TURBO-PASCAL Version 3.xx. BORLAND Inc. 1983, 1984, 1985, Heimsoeth Software, München 1985

Moderne Mikrorechnersysteme

(Teil 3)

Prof. Dr. Peter Neubert
Ralph Willem, Karsten Künne
Technische Universität Dresden,
Sektion Informatik

5. System 80386

Der Prozessor 80386 stellt eine Weiterentwicklung des 80286 dar. Es ist der erste Prozessor der Intel-Reihe mit vollständiger 32-Bit-Architektur. Von der Software her ist der 80386 weitgehend aufwärtskompatibel zum 80286, so daß existierende Programme auch auf dem 80386 weiterverwendet werden können.
Der 80386 wurde von Anfang an in CMOS-Technologie gefertigt (CHMOS III), wodurch sich die Verlustleistung gegenüber dem 80286 von drei Watt auf zwei Watt verringerte, trotz wesentlich gestiegener Komplexität (275 000 Transistoren). Auch konnte dadurch die Taktfrequenz auf 16 bzw. 20 MHz gesteigert werden. Als Gehäuse für den 80386 dient ein 132-Pin-PGA.
Der maximale physische Adreßbereich des 80386 beträgt 4 GByte, virtuell sind 64 TByte verfügbar. Ein E/A-Bereich von 64 KByte existiert ebenfalls noch.
Obwohl der 80386 erst seit 1986 verfügbar ist, hat er schon eine gewisse Bedeutung erlangt, die z. B. darin zum Ausdruck kommt, daß er in den Spitzenmodellen des neuen IBM-PS/2-Systems eingesetzt wird.

5.1. Bestandteile einer funktionsfähigen ZVE

Um eine funktionsfähige ZVE mit dem 80386 aufzubauen, gibt es verschiedene Möglichkeiten, die sich durch den Aufwand unterscheiden. So ist z. B. die Wahl zwischen einem 16 Bit breiten oder einem 32 Bit breiten Datenbus möglich. Die CPU verfügt dazu über einen Steuereingang BS16, mit dem die Datenbusbreite dynamisch von Buszyklus zu Buszyklus geändert werden kann. Dadurch ist es möglich, auch 16- und 32-Bit breite Speicher- und E/A-Einheiten gemischt zu verwenden. In diesem Fall muß allerdings eine zentrale Decodierung der Adressen vorgenommen werden, um den Steuereingang BS16 am Beginn von 16-Bit-Buszyklen zu aktivieren. Der 80386 wickelt dann den gesamten Transfer auf den niederwertigen 16 Bit des Datenbusses ab und zerlegt 32-Bit-Transfers in zwei 16-Bit-Übertragungen. Zur Anzeige, welche Bytes des Datenbusses für die Übertragung genutzt werden, gibt der 80386 außerdem noch vier Bytefreigabesignale BE0 bis BE3 aus, die zur selektiven Steuerung der Datenbustreiber genutzt werden können.
Ein weiterer Steuereingang des 80386 (NA) dient dazu, das Adreßpipelining, also die vorzeitige Adreßausgabe, an- und abzuschalten. Diese Umschaltung kann ebenfalls dynamisch von Buszyklus zu Buszyklus erfolgen. In Bild 30 sind vorzeitige und nicht vorzeitige Adreßausgabe im Taktdiagramm dargestellt.
Eine Kombination von 16-Bit-Bus und vorzeitiger Adreßausgabe ist allerdings nicht möglich. Bei Wahl der vorzeitigen Adreßausgabe können z. B. Speichersysteme mit verschachtelten Speicherbanken verwendet werden, auf die der Prozessor möglichst abwechselnd zugreift, wodurch längere Zugriffszeiten für die Speicher möglich sind. Während die Daten dann von einer Bank übertragen werden, kann die zweite schon adressiert werden (siehe Bild 31).
Ein weiterer Umstand, der den Entwurf einer ZVE beeinflußt, ist der Aufbau des Buscontrollers. Für den 80386 wird dazu kein fertiger Schaltkreis angeboten, so daß der Anwender dort variieren kann. Es ist also möglich, mit wenig Aufwand einen simplen lokalen Bus zu implementieren oder aber mit Hilfe entsprechender VLSI-Schaltkreise (z. B. 82389) und beträchtlichem Aufwand ein Multibus II-Interface zu gestalten. Als einziges existiert für den 80386 der Taktgenerator **82384**, der auch die Synchronisation des Adreßstrobe-Signals ADS und des Reset-Signals vornimmt.
Im Bild 32 ist das Blockschaltbild einer einfachen ZVE dargestellt, die an einem lokalen Bus mit 16 und 32 Bit breiten Einheiten zusammenarbeitet.
Der lokale Buscontroller hat die Aufgabe, die Statusleitungen der CPU zu decodieren und entsprechende Kommandos abzuleiten. Die Statuscodierung zeigt Tafel 6.

Tafel 6 *Statuskodierung des 80386 /9/*

M/IO#	D/C#	W/R#	Typ des Buszyklus
0	0	0	Interruptbestätigungszyklus
0	0	1	Leerzyklus
0	1	0	E/A-Lesezyklus
0	1	1	E/A-Schreibzyklus
1	0	0	Befehlsholezyklus
1	0	1	Halt- o. Abschaltungsanzeigezyklus
1	1	0	Speicherlesezyklus
1	1	1	Speicherschreibzyklus

Weiterhin synchronisiert der lokale Buscontroller das Ready-Signal, welches von Speicher- bzw. E/A-Einheiten geliefert wird. Die Adreßdecodierungslogik aktiviert den Eingang BS16, wenn die CPU auf 16-Bit-Einheiten zugreift. Das Adreßpipelining wird nicht genutzt, d. h., der Eingang NA wird nicht aktiviert. Dadurch können Adreßlatches entfallen. Bei einem größeren System sind allerdings noch Treiber für Daten- und Adreßleitungen notwendig. Die Steuerung der Datentreiber übernimmt dann ebenfalls der lokale Buscontroller.
Um die Leistungsfähigkeit der ZVE weiter zu erhöhen, kann diese noch durch einen Cachespeicher ergänzt werden, in dem bereits benutzte Daten und Befehle abgelegt werden. Der Zugriff auf diesen Cache erfolgt ohne Wartetakte. Ein einfacher Cachespeicher kann z. B. durch schnelle RAMs und Komparatoren realisiert werden. Bild 33 zeigt die Variante einer ZVE mit Cachespeicher.
Der Cachecontroller hat im wesentlichen die Aufgabe, bei Lesezugriffen die höherwertigen Adressen mit den abgespeicherten Tags zu vergleichen, um Treffer festzustellen. Bei einem Cachetreffer liefert der Daten-RAM die Daten, ansonsten wird der Hauptspeicher adressiert und die Daten werden parallel in den Cache eingeschrieben. Schreibzugriffe erfolgen gleichzeitig in Cache und Hauptspeicher.

5.2. Interruptverhalten

Das Interruptsystem des 80386 ist identisch mit dem des 80286 und 8086. Es existieren wiederum die beiden Eingänge INTR und NMI. Auch die Verwendung des Interruptcontrollers 8259A ist auf die gleiche Weise wie beim 8086 möglich.
Softwareinterrupts (INT-Befehle) gibt es ebenfalls wie beim 80286 und 8086, nur ist beim 80386 noch ein weiterer Interrupt für eine Befehlsausnahme reserviert. Interrupt 14 erfolgt, wenn ein Fehler beim Umsetzen der Linearadresse in die physische Adresse in der Seiteneinheit auftritt. Diese Befehlsausnahme ist wiederstartbar.

5.3. Die CPU 80386

Beim 80386 wurde die gesamte Architektur auf 32 Bit Verarbeitungsbreite erweitert und gleichzeitig stark verbessert. Außerdem wurde eine Demand-Paging-Speicherverwaltungseinheit (MMU) implementiert. Andere Erweiterungen und Verbesserungen betreffen den Registersatz und den Befehlssatz.

5.3.1. Architektur

Ein Blockschaltbild der internen Architektur des 80386 zeigt Bild 34.
Intern besteht der Prozessor aus drei großen Einheiten, der zentralen Verarbeitungseinheit, der Speicherverwaltungseinheit und der Buseinheit.
Die zentrale Verarbeitungseinheit ist für die Abarbeitung der Befehle verantwortlich. Sie unterteilt sich in zwei Einheiten, die Ausführungseinheit und die Befehlseinheit. Die Ausführungseinheit arbeitet die Befehle ab. Sie verfügt dazu über den allgemeinen Registersatz, eine ALU und einen Barrel-Shifter, der besonders Multiplikationen und Verschiebungen beschleunigt. Eine Multiplikation wird z. B. in weniger als einer Mikrosekunde ausgeführt. Weiterhin gehört zur Ausführungseinheit der Steuer-ROM mit dem Mikroprogramm. Die Befehlseinheit holt die Befehle aus der Codewarteschlange, decodiert sie und speichert sie in der Befehlswarteschlange ab, ähnlich der Befehlseinheit beim 80286.
Eine weitere Einheit im 80386 ist die Speicherverwaltungseinheit. Sie besteht wiederum aus Segment- und Seiteneinheit. Die Segmenteinheit übernimmt dabei die Verwaltung der Segmente und die Berechnung der Linearadresse aus der logischen Adresse. Die Seiteneinheit stellt eine Neuerung des 80386 gegenüber seinen Vorgängern dar. Sie hat die Aufgabe, aus der Linearadresse die physische Adresse zu berechnen. Dazu benutzt sie einen Zwei-Tabellen-Seiten-Mechanismus.
Die Buseinheit des 80386 gewährleistet ein vollständiges 32-Bit-Systeminterface. Bestandteil der Buseinheit ist die Codeholeeinheit, die bei unbelegtem Bus die Codewarteschlange füllt.
Alle Einheiten im 80386 sind durch mehrere interne 32-Bit-Busse verbunden und bilden eine sechsstufige Pipeline zur Befehlsabarbeitung.

5.3.2. Registersatz

Bild 35 gibt einen Überblick über den Registersatz des 80386 mit Ausnahme der Systemadreßregister.
Der Registersatz des 80386 enthält folgende Kategorien:
– allgemeine Register

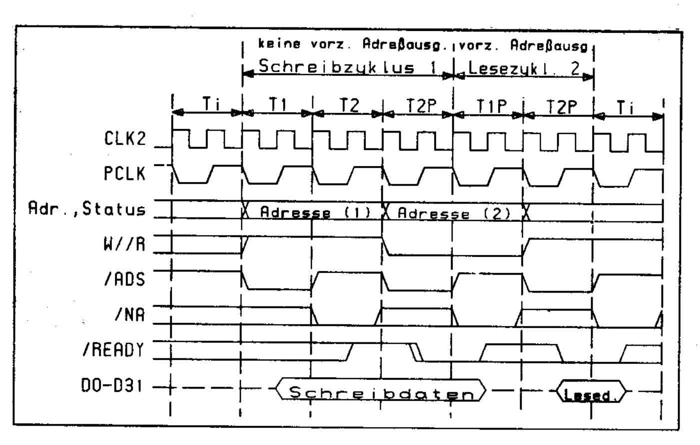


Bild 30 Taktdiagramm des 80386 /19/

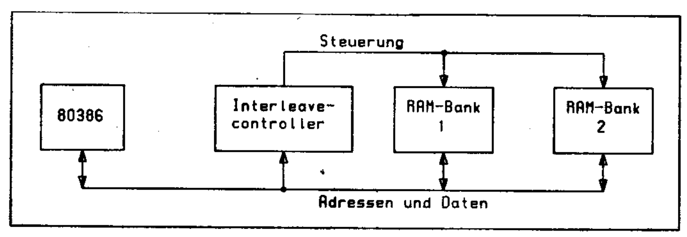


Bild 31 Speicherbankverschachtelung beim 80386 /19/

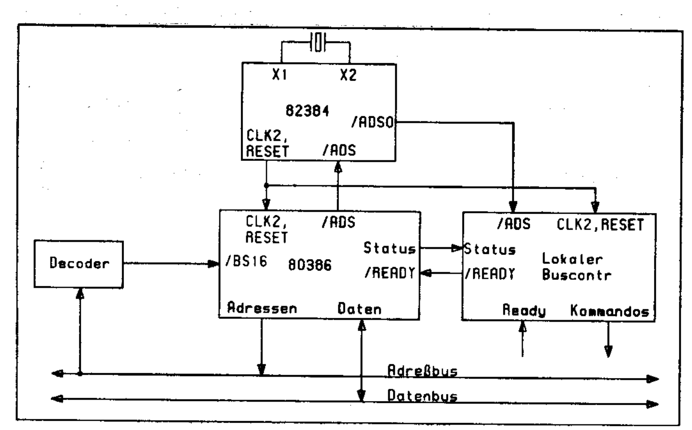
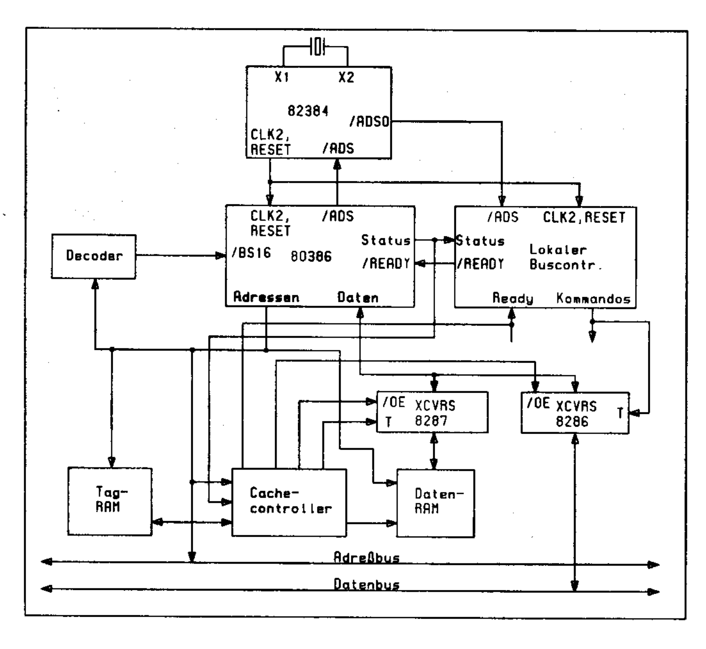


Bild 32 Einfache Variante einer ZVE mit dem 80386



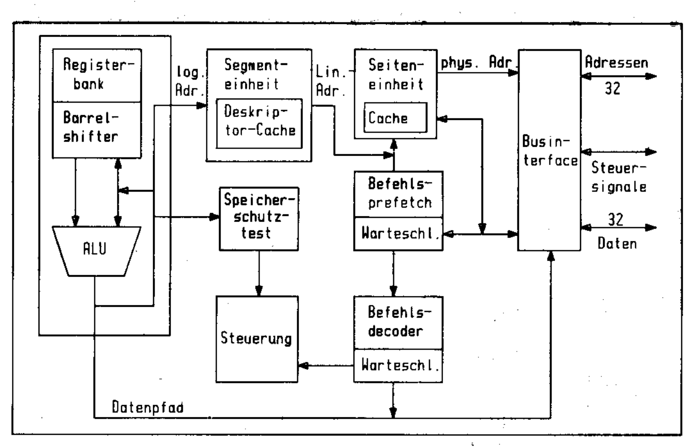


Bild 34 Blockschaltbild des 80386 /19/

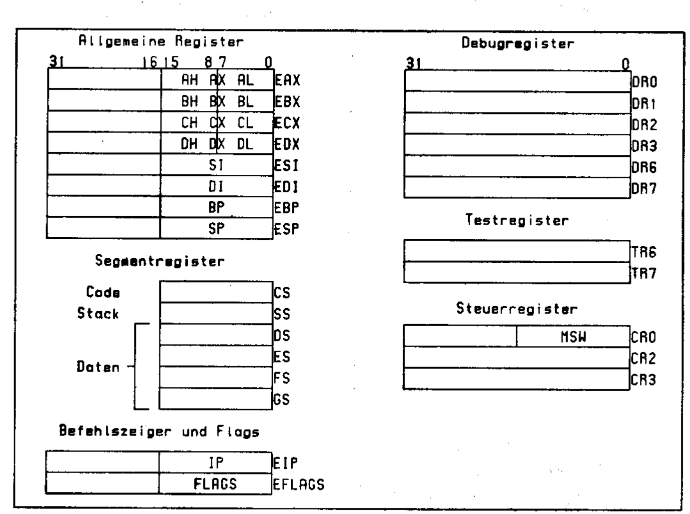


Bild 35 Registersatz des 80386 /19/

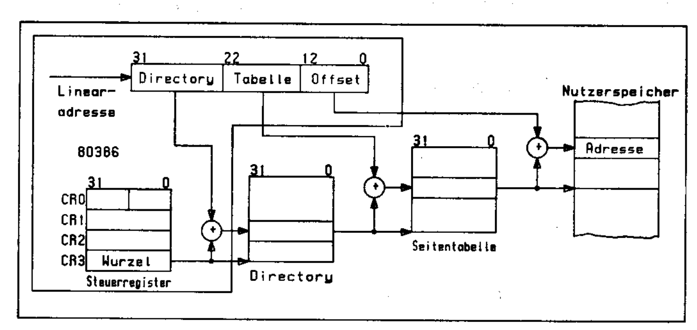


Bild 36 Seitenadressierung beim 80386 /19/

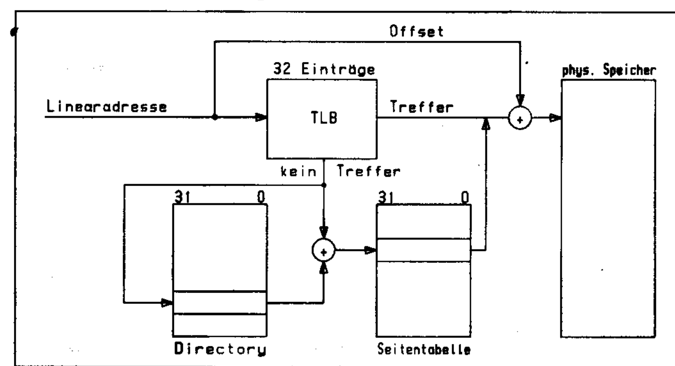


Bild 37 Funktion des TLB beim 80386 /19/

◀ *Bild 33 ZVE mit dem 80386 und einem Cachespeicher*

– Segmentregister
– Befehlszeiger und Flags
– Steuerregister
– Debugregister
– Testregister
– Systemadreßregister.

Die allgemeinen Register wurden alle auf 32 Bit Breite erweitert, können aber auch 16 Bit breit genutzt werden bzw. zum Teil als zwei 8-Bit-Register wie beim 8086. Beim 80386 ist es möglich, bei der Basisadressierung alle allgemeinen Register als Basisregister zu nutzen. Des weiteren können, bis auf ESP, auch alle Register als Indexregister für die indizierte Adressierung verwendet werden.

Die Zahl der Segmentregister wurde beim 80386 um zwei Datensegmentregister erweitert. Die Verwendung der Segmentregister unterscheidet sich nicht vom 80286.

Der Befehlszeiger wurde auf 32 Bit erweitert, ebenso das Flagregister, und es wurden zwei neue Flags gegenüber dem 80286 eingeführt:

– **VM**: Virtueller 8086-Mode; schaltet den 80386 innerhalb des Protected-Mode in den virtuellen 8086-Mode
– **RF**: Resume-Flag; wird in Verbindung mit den Haltepunktregistern benutzt, um eine Wiederaufnahme der Befehlsabarbeitung auf der Haltepunktadresse zu ermöglichen.

Die Steuerregister sind drei 32-Bit-Register, die der Prozessorsteuerung dienen. CR0 entspricht dabei dem auf 32 Bit erweiterten MSW des 80286. CR2 enthält bei Fehlern in der Seiteneinheit die Linearadresse, die den Fehler verursachte. CR3 enthält die physische Basisadresse der Seitendirectory.

Auf die Verwendung der Debug- und Textregister wird in einem gesonderten Abschnitt eingegangen.

Die Funktion der Systemadreßregister unterscheidet sich nicht vom 80286.

5.3.3. Adressierung

Auch beim 80386 muß man bei der Adressierung zwei Betriebsarten unterscheiden. Im Real-Adreß-Mode arbeitet der Prozessor als schneller 8086 mit voller Objektcodekompatibilität. Allerdings ist auch die Verwendung von 32-Bit-Operanden möglich.

Die Hauptbetriebsart des 80386 ist wie beim 80286 der Protected-Mode. Im Folgenden sollen nur die Unterschiede zwischen dem 80286 und dem 80386 im Protected-Mode betrachtet werden. Als maximaler physischer Adreßbereich stehen im Protected-Mode vier GByte zur Verfügung, virtuell 64 TByte. Es wird mit segmentiertem Speicher gearbeitet, wobei die Segmentlänge zwischen einem Byte und vier GByte liegen kann. Eine weitere Adreßumsetzung und Speicherverwaltung kann durch die Seiteneinheit vorgenommen werden. Diese ist aber auch abschaltbar.

• **Adressierungsarten**

Beim 80386 stehen dem Programmierer neben den logischen Adressierungsarten des 8086 noch drei zusätzliche zur Verfügung. Diese Adressierungsarten entstehen, wenn der Inhalt des Indexregisters mit einem Skalierungsfaktor (2,4 oder 8) multipliziert wird. Es sind im einzelnen folgende:

– *skalierte Indexadressierung*: der Indexregisterinhalt wird mit einem Skalierungsfaktor multipliziert und eine Verschiebung dazuaddiert
– *Basis-skalierte Indexadressierung:* der Indexregisterinhalt wird mit einem Skalierungsfaktor multipliziert und zum Inhalt eines Basisregisters addiert
– *Basis-skalierte Indexadressierung mit Verschiebung*: zusätzlich zur vorigen Adressierungsart wird noch eine Verschiebung addiert.

Weiterhin kann beim 80386 mit 16- oder 32-Bit-Offset gearbeitet werden. Bei Verwendung des 16-Bit-Offset ist der Prozessor in der logischen Adressierung voll kompatibel zum 8086.

• **Umwandlung der logischen in die physische Adresse**

Die logische Adresse, die der Programmierer benutzt, wird im Protected-Mode des 80386 durch zwei Einheiten, die Segment- und die Seiteneinheit, umgewandelt. Die Seiteneinheit ist auch abschaltbar. Die Segmenteinheit arbeitet ähnlich wie die des 80286 im Protected-Mode. Der reservierte Teil der Segmentdeskriptoren wird allerdings mit benutzt, und die maximale Segmentlänge kann vier GByte betragen. Es ist auch möglich, Segmente und Deskriptoren des 80286 zu benutzen, so daß vorhandene Software weiterverwendet werden kann. Der Prozessor kann 80286- und 80386-Segmente unterscheiden und richtig abarbeiten.

Die Privilegniveaus, das Schutzkonzept und die Taskumschaltung sind identisch zum 80286. Ein Taskzustandssegment vom 80386 unterscheidet sich aber von einem des 80286. Beide Typen sind verwendbar.

Die Segmenteinheit des 80386 liefert eine 32-Bit-Linearadresse, die von der Seiteneinheit weiter umgesetzt werden kann. Die Umsetzung erfolgt dabei über zwei Tabellenniveaus (siehe Bild 36).

Alle Elemente der Seitenadressierung, also Tabellen und Seiten, sind immer vier KByte lang und besitzen eine durch 4096 teilbare Basisadresse, d. h. die niederwertigen 12 Bit der Basisadresse sind Null.

Die erste Umsetzungstabelle ist die Seitendirectory. Sie enthält maximal 1024 Einträge. Da jede Task eine eigene Seitendirectory benutzen kann, ist hier wiederum eine Taskisolation möglich. Die Bits A22 bis A31 der Linearadresse werden als Index in der Seitendirectory benutzt und wählen einen Eintrag aus. Dieser Eintrag enthält dann die Basisadresse einer Seitentabelle.

Die Seitentabelle ist die zweite Umsetzungstabelle. Sie kann wiederum bis zu 1024 Einträgen enthalten. Die Bits A12 bis A21 der Linearadresse werden als Index in der Seitentabelle benutzt und wählen wieder einen Eintrag aus, der dann die höherwertigen 20 Bit der physischen Adresse liefert. Die niederwertigen 12 Bit der physischen Adresse werden aus der Linearadresse übernommen.

Auch auf dem Niveau der Seitenadressierung existiert beim 80386 ein Zugriffsschutz, der allerdings nur zwei Privilegniveaus unterscheidet, User und Supervisor. Weiterhin können einzelne Seiten auch schreibgeschützt werden.

Um die Adreßumsetzung zu beschleunigen, besitzt die Seiteneinheit einen Cachespeicher, den Translation-Lookaside-Buffer (TLB). Der TLB enthält die Adreßzuordnung für die 32 meistbenutzten Seiten. Dadurch werden 128 KByte Speicheradreßbereich erfaßt, was für die lokale Umgebung eines Programms ausreicht. Bild 37 veranschaulicht die Funktion des TLB.

• **Virtueller 8086-Mode**

Innerhalb des Protected-Mode ist es beim 80386 möglich, 8086-Programme in eigenen Tasks unter vollem Speicherschutz abzuarbeiten. Der Prozessor verhält sich dabei wie ein 8086.

Der Adreßbereich der virtuellen 8086-Tasks von einem MByte kann durch die Seiteneinheit auf den physischen Adreßbereich des 80386 von vier GByte aufgeteilt werden. Jede virtuelle 8086-Task kann dabei maximal 256 Seiten belegen. Der 80386 ist somit in der Lage, mehrere 8086-Prozessoren gleichzeitig zu emulieren.

Eine virtuelle 8086-Task arbeitet immer auf dem niedrigsten Privilegniveau. Dadurch ist es möglich, z. B. E/A-Befehle, Interruptbefehle und Flagmanipulationen durch das 80386-Betriebssystem aufzufangen und zu emulieren. Somit kann auch die Umgebung der virtuellen 8086-Tasks vollständig an die Erfordernisse der Software angepaßt werden.

Das Eintreten in den virtuellen 8086-Mode und das Verlassen desselben erfolgen einfach durch eine Taskumschaltung.

Der virtuelle 8086-Mode gestattet die Integration von 8086-Software in ein Protected-Mode-80386-System mit allen seinen Vorzügen.

5.3.4. Befehlssatz

Gegenüber dem 80286 wurde der Befehlssatz des 80386 erweitert. Es wurden besonders Befehle zur Unterstützung von Multitaskingbetriebssystemen neu eingeführt.

Eine erste Gruppe neuer Befehle sind Bitmanipulationsbefehle. Mit diesen Befehlen lassen sich einzelne Bits testen, setzen, rücksetzen bzw. negieren. Beispiele sind BTS (Bit testen und setzen) oder BTC (Bit testen und negieren). Zwischen den beiden Teiloperationen z. B. des BTS-Befehls wird der Bus verriegelt (LOCK-Signal), so daß kein anderer Busmaster zwischendurch Zugriff auf das Bit erhalten kann.

Die zweite neue Befehlsgruppe sind bedingte Bytesetzbefehle. Damit lassen sich in Abhängigkeit von Bedingungen bestimmte Bytes setzen. Zu diesen Befehlen gehören z. B. SETLE (Setze Byte bei kleiner oder gleich) oder SETNE (Setze Byte bei ungleich).

Als letzte neue Befehlsgruppe seien noch die Befehle erwähnt, die dazu dienen, die neuen Debug-, Steuer- und Testregister zu laden bzw. abzuspeichern.

Neben der Einführung neuer Befehle wurde auch der vom 80286 übernommene Befehlssatz erweitert. Diese Erweiterung erfolgte hauptsächlich durch die Möglichkeit, 32-Bit-Operanden und 32-Bit-Adressierungsarten zu verwenden. Als allgemeine Regel gilt, daß bei 80386-Code die Operanden 8 oder 32 Bit lang sind und ein 32-Bit-Offset verwendet wird, bei 80286-Code dagegen sind die Operanden 8 oder 16 Bit lang und der Offset 16 Bit. Diese allgemeine Regel kann aber durch Befehlspräfixe für Operanden- und Adreßlänge übergangen werden. Die Präfixe haben dann in 80386- und 80286-Segmenten jeweils entgegengesetzte Bedeutung. Auf Assemblerniveau ist die genannte Regel analog. Der Programmierer kann also alle Operanden- und Adreßlängen mischen. Notwendige Präfixe werden durch den Assembler automatisch generiert.

5.4. Debug- und Testunterstützung

Gegenüber seinen Vorgängern wurden beim 80386 die Möglichkeiten für verschiedene Tests und die Unterstützung beim Softwareentwurf wesentlich erweitert.

Als erste neue Testmöglichkeit ist da der Selbsttest zu nennen. Der Selbsttest prüft die Funktion des größten Teils der festverdrahte-

ten Logik und den gesamten Mikrocode-ROM. Das ist etwa die Hälfte des Prozessors. Als Testverfahren dient dabei die Signaturanalyse. An die Logik werden bestimmte Eingangsfolgen gelegt. Aus den Ausgangsfolgen wird dann in einem Register eine Signatur berechnet und diese mit einer abgespeicherten verglichen. Der Selbsttest wird durch das Potential einer Leitung an der Rückflanke des *Reset*-Impulses ausgelöst und benötigt etwa 30 ms /18/. Nach Beendigung des Selbsttests führt der Prozessor *Reset* durch und beginnt die Abarbeitung des Programms. Wenn die Inhalte der Register EAX und EDX Null sind, war der Selbsttest erfolgreich.
Eine weitere Testmöglichkeit beim 80386 ist der Test des TLB. Die Seiteneinheit muß bei diesem Test abgeschaltet werden. Mit Hilfe der beiden Testregister TR6 und TR7 ist es dann möglich, ein Testmuster in den TLB zu schreiben und Zugriffe zu simulieren.
Als neue Debug-Unterstützung gibt es beim 80386 die Möglichkeit, hardwaremäßig Haltepunkte zu setzen. Das erfolgt mit Hilfe der sechs Debugregister. Es können maximal vier Haltepunkte gesetzt werden. Diese Haltepunkte können nicht nur Befehlshaltepunkte sein, sonden es ist auch möglich, Haltepunkte bei Datenzugriffen zu setzen. Das kann z. B. genutzt werden, um zu prüfen, an welcher Stelle in einem Programm eine bestimmte Variable geändert wird. Weiterhin können die Haltepunkte global oder lokal für eine Task gesetzt werden.
Die vom 8086 bekannte Einzelschrittmöglichkeit für Software mit Hilfe des Trap-Flag exisitiert beim 80386 weiterhin.
Alle diese Debug-Hilfen auf Hardwareniveau vereinfachen die Entwicklung von Debuggern und ähnlichen Softwarewerkzeugen sicher wesentlich.

Schluß

Literatur

/18/ El-Ayat, K. A.; Agarwal, R. K.: The Intel 80386-Architecture and Implementation. IEEE Micro Vol. 5/Dezember 1985
/19/ 80386 High Performance Microprocessor with Integrated Memory Management. Intel Corporation 1985
/20/ 82384 Clock Generator and Reset Interface for 80386 Processors. Intel Corporation 1985
/21/ Duzy, P.; Schallenberger, B.; Wallstab, S.: Moderne Mikroprozessoren. Elektronik 23 (1986) 14

Wegbereiter der Informatik



SIR ISAAK NEWTON

∗ 1643 Woolsthorpe,
+ 1727 Kensington bei London.

Newton studierte von 1661 bis 1665 am Trinity-College in Cambridge. Diese Bildungseinrichtung war damals noch im wesentlichen eine mittelalterlich organisierte Universität, d. h. die alten Sprachen und theologische Fächer spielten zunächst die Hauptrolle. Newton hat aber – nach eigenen Angaben – in den ersten Studienjahren auch die Werke Euklids studiert und das kopernikanische Weltsystem kennengelernt. Sein Lehrer in Mathematik und Naturwissenschaften war der Mathematiker und Theologe I. Barrow als Inhaber des einzigen, erst 1663 eingerichteten naturwissenschaftlichen Lehrstuhls von Cambridge. Nach Erwerb verschiedener akademischer Grade wurde Newton als Nachfolger seines Lehrers Professor für Mathematik. Dieses Lehramt bekleidete er bis 1701; in dieser Periode liegen seine schaffensreichsten Jahre. Da er durch eine Feuersbrunst (1693) sein Laboratorium und einen Teil seiner Manuskripte verlor, zog er sich danach von der Wissenschaft fast vollständig zurück und befaßte sich als Aufseher der Münze (1695) mit dem Aufbau des englischen Münzwesens; 1699 wurde Newton Direktor der Londoner Münze. Im Jahre 1703 siedelte er nach London über und wurde Präsident der Royal Society, der er seit 1671 bereits als Mitglied angehörte.
Newtons Lebenswerk umfaßt neben theologischen und alchemistischen Schriften vor allem Arbeiten auf den Gebieten der Mathematik, der experimentellen Optik und der theoretischen Mechanik.
In der Mathematik hat er eine Reihe praktikabler numerischer Verfahren entwickelt, die heute in der Regel zur Grundausstattung einer Computer-Software gehören. Genannt werden soll da zum einen das Newtonsche Iterationsverfahren zur Auflösung von Gleichungen mit einer Unbekannten:

$$Z_{i+1} = Z_i - F(z_i) \cdot F'(z_i)^{-1};$$

es handelt sich um ein überlinear konvergentes Verfahren, das auch gern zur Nullstellenberechnung für Polynome höheren Grades herangezogen wird.
Auch die bekannte Drei-Achtel-Regel stammt von ihm, eine Näherungsformel zur Berechnung des bestimmten Integrals einer Funktion:

$$\int_0^{3h} y(x)dx = (y_0 + 3y_1 + 3y_2 + 3y_3) \cdot 3h/8.$$

Sie hat die gleiche Genauigkeitsordnung wie die Keplersche Faßregel, Newton soll sie Pulcherrima – die Schönste – genannt haben. Des weiteren verdanken wir ihm einen Iterationsalgorithmus zur Berechnung von Quadratwurzeln sowie eine Formel zur Interpolation von Funktionen.
Sich auf Arbeiten von B. Cavallieri, P. Fermat und I. Barrow stützend, entwickelte er bis 1671 – unabhängig von Leibniz – als mathematischen Hilfsapparat für seine physikalischen Untersuchungen die Grundideen der Infinitesimalrechnung, die er Fluxionsrechnung nannte (Methode of Fluxions, 1736 gedruckt). Seine darin benutzte Symbolik hat sich jedoch nicht durchgesetzt, in den Naturwissenschaften hat sich der von Leibniz vorgeschlagene Kalkül als zweckmäßiger erwiesen und wird heute allgemein verwendet.
In der Optik entdeckte Newton die Spektralzerlegung des weißen Lichts (1672) und die nach ihm benannten Interferenzerscheinungen (Newtonsche Ringe, 1675); auch konstruierte er das erste brauchbare Spiegelteleskop, für welches er eigenhändig die Spiegel hergestellt hatte! Eine zusammenfassende Darstellung seiner optischen Untersuchungen veröffentlichte er aber erst 1704 nach dem Tode von R. Hooke, da er mit diesem in dauernde Prioritätsstreitigkeiten verwickelt war.
Sein für den Fortschritt der physikalischen und astronomischen Forschung wichtigstes Werk sind die „Philosophiae naturalis principia mathematica" (Mathematische Prinzipien der Naturlehre, 1687). Darin faßte er die Leistungen seiner Vorgänger Galilei, Kepler, Huygens, v. Guericke u. a. zusammen und stellte das Gravitationsgesetz auf, mit dem er eine mathematische Theorie zur Erklärung der Bewegung der Himmelskörper schuf und die drei Keplerschen Gesetze rechtfertigte. Mit der Formulierung der drei (Newtonschen) Axiome der Mechanik vollendete er schließlich den Aufbau der klassischen Mechnik.
Newton publizierte seine wissenschaftlichen Ergebnisse sehr ungern. Als er seine Fluxionsrechnung allgemein bekannt machte, war seine Art der Behandlung von Problemen der Analysis gegenüber dem Kalkül von Leibniz bereits veraltet. Um so befremdlicher berühren uns heute die Anschuldigungen, die Newton und besonders seine Anhänger gegenüber Leibniz erhoben, indem sie diesen bezüglich der Erfindung der Infinitesimalrechnung des Plagiats bezichtigten und damit einen unliebsamen und lang andauernden Prioritätsstreit auslösten. Leibniz' Versuch, dieserhalb mit Newton in direkten Briefwechsel zu treten, endete mit einer abweisenden Antwort (1693). Leibniz ging zwar aus diesem Streit schließlich in dem Sinne als Sieger hervor, als sich die gegen ihn gerichteten Anschuldigungen als völlig ungerechtfertigt erwiesen haben, Newton hatte sich mit seiner Verdächtigung geirrt, da er die ihm zugänglichen Unterlagen nicht hinreichend sorgfältig angesehen hat.
Newton erhielt nach seinem Tode ein Staatsbegräbnis und wurde in der Westminster-Abbey beigesetzt; sein Sarkophag wurde mit der von ihm gefundenen Formel für die Binominalreihe graviert.

Dr. Klaus Biener

PASCAL (Teil 5)

Dr. Klaus Kofer
Informatikzentrum des Hochschulwesens an der Technischen Universität Dresden

9.4. Programmbeispiel

Mengentypen können vorteilhaft dann angewendet werden, wenn im Programm bestimmte Entscheidungen davon abhängen, ob eine Variable irgendeinen aus einer Menge von möglichen Werten hat.
Falls in dem bereits gezeigten Programm zur Ermittlung der Häufigkeit einzelner Zeichen in einem Textfile nur die Anzahl der Vokale ermittelt werden soll, könnte die Lösung ohne Verwendung von Mengen so aussehen:

```
IF(C = 'A') OR (C = 'a')  OR
  (C = 'E') OR (C = 'e')  OR
  (C = 'I') OR (C = 'i')  OR
  (C = 'O') OR (C = 'o')  OR
  (C = 'U') OR (C = 'u')  THEN...
```

Eleganter ist die Lösung mit Mengen:

```
TYPE     CharSet = SET OF CHAR;
VAR      Vokale: Char Set;
...
Vokale: =   ['A', 'E', 'I', 'O', 'U',
             'a', 'e', 'i', 'o', 'u'];
...
...
IF G IN Vokale THEN...
...
```

Die Konstruktion der Menge Vokale braucht natürlich nur einmal am Beginn der Programmabarbeitung durchgeführt werden.

9.5. Interne Darstellung

Die interne Darstellung einer Menge ist ein Bitmuster. Die Arbeit mit Mengen entspricht der Manipulation von Bitmustern. Durch Kenntnis der internen Abläufe bei der Arbeit mit Mengen können möglicherweise weitere sinnvolle Anwendungen des Mengentyps erschlossen werden.
Durch die Deklaration

```
VAR M:SET OF(W0, W1, W2, ..., Wn);
```

plant der Compiler für die Variable M eine Anzahl von Bits. Gewöhnlich ein Vielfaches von acht, mindestens aber n+1. Die Bits werden durchnumeriert 0, 1, 2, ..., n−1, n. Nun korrespondiert jedes Bit mit einem der Werte W0, W1, ..., Wn:

```
W0 – BIT 0
W1 – Bit 1
...
Wn – Bit n
```

Gehört das Element Wi der Menge an, so hat das korrespondierende Bit i den Wert 1, sonst 0.
Aufgrund dieser Darstellung können die Mengenoperationen unmittelbar durch logische Befehle des Maschinenbefehlssystems ausgeführt werden. Bit-Setze- und Bit-Test-Befehle bewerkstelligen die Konstruktion von Mengen und den Test, ob ein Element in einer Menge enthalten ist.
Die Nützlichkeit der Mengen hängt wesentlich davon ab, wie differenziert der Compiler die Zuordnung von Speicherplatz vornimmt. Relativ unbrauchbar ist sicher eine Realisierung für Mengen, die unabhängig davon, wieviele Elemente der Wertebereich des Basistyps tatsächlich umfaßt, stets die maximale Anzahl von 64, 128 oder 256 Bits (entsprechend 8, 16 oder 32 Byte) zuordnet.
Bei TURBO-PASCAL werden nur so viele Bits geplant, wie der Basistyp Elemente hat. Dabei wird auf das volle Byte aufgerundet.

10. Datentyp Pointer

10.1. Einführung

Die Variablendeklaration veranlaßt den Compiler, Speicherplatz entsprechend dem Bedarf des jeweiligen Datentyps zu planen.
Es gibt jedoch Programme, bei denen sich erst während ihrer Arbeit – etwa in Abhängigkeit von den Eingabedaten – Menge und Typ des erforderlichen Speichers für Daten herausstellt. Hierfür ist ein Mechanismus wünschenswert, der dem Programm die „eigenverantwortliche" Abforderung vom Datenspeicher gestattet. Pointertypen ermöglichen das.

10.2. Syntax

Der Pointertyp ist eine Alternative im Syntaxdiagramm „typ" in Bild 3.1. Seine Deklaration zeigt Bild 10.1. Der Bezeichnung muß der Name eines Datentyps sein. Das Zeichen "^" ist zu lesen wie „Zeiger auf". Pointer verweisen also auf Datenobjekte eines bestimmten, bei der Deklaration festgelegten Typs. Der Wert einer Pointervariablen ist stets eine Hauptspeicheradresse. Der Datentyp, auf den sie verweist, hängt von ihrer Deklaration ab. Nachfolgend werden ein Pointer auf INTEGER, ein Pointer auf ein Array mit vier Komponenten vom Typ REAL und ein Pointer auf einen Record mit zwei Feldern der Typen REAL und CHAR gezeigt:

```
TYPE IntPtr = ^ INTEGER;
ArrPtr = ^ ARRAY [0...3]
                    OF REAL;
RecPtr = ^ RECORD
              FA: REAL;
              FB: CHAR;
           END;
```

Obwohl die Werte aller drei Pointervariablen Hauptspeicheradressen sind, sind sie nicht miteinander verträglich. Denn sie verweisen auf Datenobjekte ganz unterschiedlichen Typs.

→ ^ → bezeichner →

Bild 10.1 Syntaxdiagramm „pointertyp"

Zum Zugriff auf die durch Pointer referenzierten Datenobjekte wird der Pointervariablen das Zeichen "^" nachgestellt.
Falls die Variablen

```
VAR PI: IntPtr;
    PA: ArrPtr;
    RR: RecPtr;
```

deklariert sind, stellt PI^ ein Datenobjekt vom Typ INTEGER, PA^ ein Array und PR^ einen Record dar. Mit diesen Daten dürfen alle Operationen durchgeführt werden, die für ihre Typen erlaubt sind, z. B.:

```
2*PI^ + 1
PA^[2] + 0.5
PR^.FA + 1.0
```

Der Deklarationszwang von PASCAL erfordert es, Bezeichner vor ihrer ersten Verwendung zu deklarieren. Beim Pointertyp gibt es die einzige Ausnahme. Sie ist notwendig, damit gleiche Datenobjekte miteinander „verzeigert" werden können.
Das Beispiel

```
TYPE Zeiger = ^ Knoten;
     Knoten = RECORD
       Info:...;
       Nachfolger: Zeiger
     END;
```

kann ohne Vorgriff nicht gelöst werden.
Falls der Datentyp Zeiger nicht noch in anderen Deklarationen benötigt wird, kann auch geschrieben werden:

```
TYPE Knoten = RECORD
       Info:...;
       Nachfolger: ^ Knoten
     END;
```

Auch hier wird bereits auf Knoten Bezug genommen, obwohl seine Deklaration noch nicht beendet ist.

10.3. Arbeit mit Pointern

Für Pointervariablen gleichen Typs ist die Zuweisung und die Ausführung der Vergleichsoperationen = und <> erlaubt. Mit dem Datenobjekt, auf das ein Pointer verweist, können die für seinen Datentyp zulässigen Operationen durchgeführt werden.
Es gibt die vordeklarierte Konstante NIL, die mit allen Pointertypen verträglich ist. Falls ein Pointer den Wert NIL hat, so zeigt er auf kein Datenelement. Der Zugriff über eine Pointervariable, die den Wert NIL hat, ist ein Fehler und führt zum Programmabbruch.
Für die Arbeit mit Pointer ist die Standardprozedur

```
NEW (p)
```

notwendig. Ihr aktueller Partner ist eine Pointervariable beliebigen Typs. Durch NEW wird für ein durch die Pointervariable referenziertes Datenobjekt Speicher bereitgestellt. Seine Adresse wird zum Wert der Pointervariablen. Den bereitzustellenden Spei-

cher entnimmt NEW dem sogenannten Heap.
Das folgende Programmstück zeigt ein Beispiel:

```
TYPE T = ARRAY [1 . . . 10] OF REAL;
VAR P:^T;
BEGIN
 . . .
 NEW(p);
 P^[1]: = 1.5;
 . . .
```

Vor dem Aufruf von NEW darf über P^ nicht zugegriffen werden. Erst durch NEW wird ein Array mit 10 REAL-Zahlen im Heap angelegt und kann anschließend genutzt werden.
Ob eine Pointervariable am Programmstart mit dem Wert NIL initialisiert ist, hängt vom verwendeten PASCAL-System ab. In TURBO-PASCAL ist ihr Wert unbestimmt. Durch die ersten Anweisungen eines Programms sollte Pointervariablen deshalb generell NIL zugewiesen werden.
Die nachfolgende Prozedur zeigt das Verzeigern von Datenobjekten zu einer Liste.

```
TYPE Zeiger = ^Knoten;
      Knoten = RECORD
       Info: Infotyp;
       Nachf: Zeiger
      END;

PROCEDURE Einf(V: Infotyp;
VAR P: Zeiger;
BEGIN
 NEW(p);
 WITH P^ DO BEGIN
  Info: = V;
  Nachf: = Anfang
 END;
 Anfang: = P
END;
```

Die einzelnen Elemente der Liste sind vom Typ des schon bekannten Records Knoten. Das Feld Info enthält die Daten. Nachf zeigt auf das folgende Listenelement. Das Einfügen eines Elements geschieht am Listenanfang und wird durch die Prozedur Einf ausgeführt. Sie erhält dazu als Parameter die einzufügende Information und einen Zeiger auf den Listenanfang. Da der Listenanfang modifiziert wird, muß er als Referenzparameter übergeben werden. Zuerst wird ein neues Listenelement angelegt. Danach wird die Information eingetragen und die bisherige Liste als Nachfolger angehängt. Anschließend wird der Anfang auf das eben angelegte Element gestellt.
Bei Variantenrecords ist es möglich, aus dem Heap nur soviel Speicher auszufassen, wie tatsächlich benötigt wird. Dazu erhält NEW weitere Parameter:

NEW (p, t1, t2, . . .);

Die ti repräsentieren die Werte der Anzeigefelder der anzulegenden Variante.
Dazu ein Beispiel:

```
VAR p:^RECORD
   CASE BOOLEAN OF
   FALSE: (C: CHAR);
   TRUE: (R: ARRAY [1 . . . 10] OF REAL)
  END;
```

Beim Aufruf von

NEW (p, FALSE)

wird nur ein Datenelement vom Typ CHAR im Heap angelegt, bei

NEW (p, TRUE)

ein Array von 10 REAL-Zahlen.
In TURBO-PASCAL können bei NEW keine Werte für Anzeigefelder angegeben werden. NEW faßt immer den maximalen Speicherplatz aus.

10.4. Programmbeispiel

Das Programm löst die Aufgabe, Namen entsprechend ihrer lexikographischen Ordnung zu sortieren. Dazu wird mit den Namen eine Baumstruktur aufgebaut. Folgender Record bildet einen Knoten dieses Baumes:

```
TYPE Knoten = RECORD
      Name: Alpha;
      Vorg: ^Knoten;
      Nachf: ^Knoten
     END;
```

Das erste Feld trägt den Namen. Die Felder Vorg und Nachf verweisen auf weitere Knoten, die so in den Baum eingeordnet sind, daß

Vorg^.Name < Name < Nachf^.Name

gilt. Falls es keine Vorgänger und/oder Nachfolger gibt, hat das entsprechende Feld den Wert NIL. Das entsprechende Programm wird im Bild 10.2 gezeigt. Zu Beginn der Arbeit ist der Baum leer, das heißt, der auf ihn verweisende Zeiger hat den Wert NIL. Durch die Prozedur Insert wird der Baum Knoten für Knoten aufgebaut. Sie steigt zunächst solange im Baum ab; bis eine leere Stelle gefunden wird, an die der einzuordnende Name entsprechend seiner lexikographischen Ordnung paßt. Dort fügt sie einen neuen Knoten ein.

```
TYPE
  alpha=ARRAY[1..10] OF CHAR;
  link=^node;
  node=RECORD
    name: alpha;
    Vorg: link;
    Nachf:link
  END;


PROCEDURE Insert(VAR Teilbaum:link; Info:alpha);
BEGIN
  IF Teilbaum=NIL THEN BEGIN
    NEW(Teilbaum);
    WITH Teilbaum^ DO BEGIN
      name:=Info; Vorg:=NIL; Nachf:=NIL
    END
  END ELSE IF Info<Teilbaum^.name THEN
    Insert(Teilbaum^.Vorg,Info)
  ELSE
    Insert(Teilbaum^.Nachf,Info)
END;

PROCEDURE Print(Teilbaum:link);
BEGIN
  IF Teilbaum<>NIL THEN BEGIN
    Print(Teilbaum^.Vorg);
    WRITELN(Teilbaum^.name);
    Print(Teilbaum^.Nachf)
  END
END;
```

Bild 10.2 Prozeduren zur Arbeit mit einem binären Baum

Die Prozedur Print traversiert den Baum. Sie druckt für jeden Knoten zuerst die Namensfelder der Vorgängerknoten, dann den Namen des Parameterknotens und danach die Namen der Nachfolgeknoten.
Die Prozeduren Insert und Print rufen sich rekursiv auf. In TURBO-PASCAL müssen sie deshalb mit der Compileroption ○A- übersetzt werden.

10.5. Verwaltung des Heap-Speichers

Durch weitere Standardprozeduren ist es möglich, nicht mehr benötigten Speicherplatz wieder in den Heap einzugliedern. Er kann so im Zuge der Abarbeitung des Programms wieder für andere Datenobjekte benutzt werden.
Zur effektiven Anwendung dieser Prozeduren ist es erforderlich, die Anordnung eines PASCAL-Programms im Hauptspeicher zu kennen. Sie wird im Bild 10.3 gezeigt. Aus prinzipiellen Gründen gibt es zwischen den PASCAL-Systemen kaum Unterschiede. Der Speicherbereich untergliedert sich zunächst in Kode und Daten. Der Kodebereich enthält das Hauptprogramm und die Prozeduren.
Der Datenbereich untergliedert sich weiter in Heap. Stack und Hauptprogrammvariable.

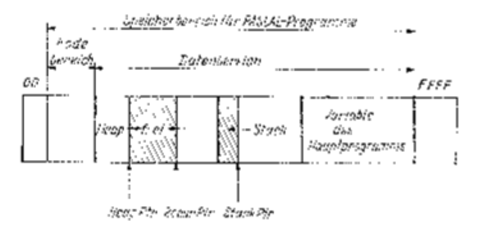


Bild 10.3 Prinzipielle Anordnung eines PASCAL-Programms im Hauptspeicher

Im Stack werden Parameter für Prozeduren und Funktionen sowie Zwischenresultate der Ausdrucksberechnung abgelegt.
Der Heap wächst nach steigenden, der Stack nach fallenden Adressen. Die PASCAL-Systeme überwachen die Ausdehnung von Stack und Heap. Ein Ineinanderlaufen von Stack und Heap führt zum Programmabbruch mit einer entsprechenden Fehlermeldung.
Bei TURBO-PASCAL gibt es noch einen Rekursionsstack, der mit Hilfe von RecurPtr verwaltet wird. In ihm sichern die rekursiv aufrufbaren Prozeduren die „alten" Werte ihrer lokalen Variablen. In TURBO-PASCAL sind HeapPtr, RecurPtr und StackPtr vordeklarierte Variablen, denen das PASCAL-Programm Werte zuweisen kann.
Ein fehlerfreies Arbeiten ist aber nur dann gewährleistet, wenn stets gilt:

HeapPtr < RecurPtr < StackPtr

Die Standardprozedur

DISPOSE(p)

gibt den Speicherplatz wieder frei, auf den die Pointervariable p verweist. Das eigentliche Problem bei DISPOSE liegt in der sinnvollen Wiederverwendung von freien Speicherbereichen, die sich inmitten des Heap befinden.

Beim TURBO-PASCAL wird darüber Buch geführt. Die Standardprozedur NEW versucht zuerst, das neu anzulegende Datenobjekt in einer Lücke zu plazieren. Die Standardfunktionen MEMAVAIL und MAXAVAIL liefern die Menge des insgesamt feien Speichers in Byte. Die Standardprozeduren

MARK (p) und
RELEASE (p)

eröffnen dem Programmierer den Zugang zu einer eigenen Heapverwaltung. MARK (p) legt den augenblicklichen Wert von HeapPtr in der Pointervariablen p ab. Durch RELEASE (p) wird HeapPtr wieder auf den Wert der Pointervariablen p gesetzt.
Es unterliegt der Verantwortung des Programmierers, zu entscheiden, wann HeapPtr zurückgesetzt wird.
Die Prozeduren MARK und RELEASE stellen die einfachste Form der Heap-Verwaltung dar. Sie gibt es in nahezu allen PASCAL-Systemen.
TURBO-PASCAL hat weiterhin die Prozeduren

GETMEM (p, n) und
FREEMEM (p, n).

Sie fassen aus dem Heap n Bytes aus bzw. geben ihn zurück. Mit GETMEM läßt sich der Mangel von NEW für Variantenrecords mindern.

11. TURBO-PASCAL-Sytem

11.1. Einführung

Gegenstand dieses Kapitels ist die Darstellung von *Erweiterungen und Leistungen* des TURBO-PASCAL-Systems, die nicht Bestandteil von Standard-PASCAL sind und in dieser Form nicht oder nur teilweise in anderen PASCAL-Systemen anzutreffen sind. Das sind:
- Typenübertragung
- typisierte Konstanten
- Datentypen BYTE und STRING
- absolute Variablen
- vordeklarierte Arrays für Zugriff auf den Hauptspeicher und die E/A-Ports
- Einfügen von Maschinenkodepassagen
- Aufruf von Maschinenkodeunterprogrammen
- Überlagerungsstrukturen für PASCAL-Prozeduren
- unterschiedliche Modi der Objektkodeerzeugung
- Quellprogrammuntergliederung.

Weiterhin wird im letzten Abschnitt die Bedienung des Systems erläutert.

11.2. Typenübertragung

Die Möglichkeit der Typenübertragung gestattet es dem Programmierer, gezielt die *Typenkontrolle des Compilers außer Kraft* zu setzen und in bestimmtem Umfang normalerweise unverträgliche Datenobjekte einander zuzuweisen.
Die Anwendung dieses Mechanismus wird an zwei Beispielen gezeigt. Im ersten Beispiel wird ein Datenelement vom Typ BOOLEAN einem INTEGER-Datenelement zugewiesen:

```
VAR i: INTEGER;
    b: BOOLEAN;
...
i:= INTEGER(b);
```

Das zweite Beispiel zeigt die Anwendung bei Aufzählungstypen:

```
TYPE farbe = (rot, blau, grün);
VAR f: farbe; i: INTEGER;
...
f: = farbe(0);
```

Entsprechend der internen Darstellung von Aufzählungstypen ist die letzte Zuweisung mit

```
f: = rot
```

äquivalent.
Bei der Typenübertragung wird nicht konvertiert. Das heißt, die Typenübertragung z. B. zwischen REAL und INTEGER ist nicht möglich.

11.3. Typisierte Konstanten

Ein oft diskutierter Mangel an Standard-PASCAL ist das Fehlen von Ausdrucksmitteln, mit denen *strukturierte Konstanten* deklariert werden können.
TURBO-PASCAL stellt zur Lösung dieses Problems sogenannte typisierte Konstanten bereit.
Mit Hilfe der nach Bild 11.1 modifizierten Konstantendeklaration werden sie in die Sprache eingebaut. Die Bilder 11.2 bis 11.5 zeigen weitere Einzelheiten der Syntax.
Eine naheliegende Anwendung könnte folgende Konstantendeklaration sein:

```
TYPE Tag = (Mo, Di, Mi,
            Do, Fr, Sa, So);
CONST WoAnfang: Tag = Mo;
```

Weitaus wichtiger dürfte aber die Deklaration von Array-, Record- und Mengenkonstanten sein. Es folgen Beispiele:



Bild 11.1 Modifiziertes Syntaxdiagramm „konstantendeklaration

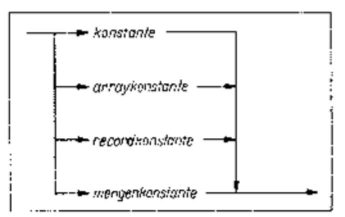


Bild 11.2 Syntaxdiagramm „strukturierte konstante"

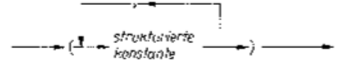


Bild 11.3 Syntaxdiagramm „arraykonstante"

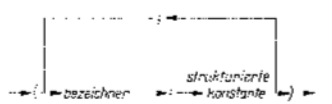


Bild 11.4 Syntaxdiagramm „recordkonstante"

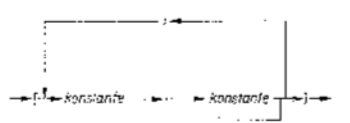


Bild 11.5 Syntaxdiagramm „mengenkonstante"

a) Deklaration von Arraykonstanten

```
TYPE Vekt = ARRAY [1...2] OF REAL;
     Mat = ARRAY [1...3, 1..2]
                              OF REAL;
CONST X: Vekt = (1.0, 2.0);
      A: Mat  = ((1.1, 1.2),
                 (2.1, 2.2),
                 (3.1, 3.2));
```

Die Werte der Arraykomponenten müssen zeilenweise aufgeführt werden.

b) Deklaration von Recordkonstanten

```
TYPE REC = RECORD
      F1: INTEGER;
      F2: CHAR
     END;
CONST R: REC = (F1:1; F2: 'A');
```

Die Recordfelder werden durch Voranstellen ihrer Bezeichner gekennzeichnet.

c) Mengenkonstanten

```
TYPE S = SET OF (A, B, C);
CONST S1: S = [A, C];
      S2: S = [A];
```

Die nun folgenden Beispiele zeigen die Kombination der Grundelemente:

```
TYPE T1 = ARRAY [1..2] OF
          RECORD
            F1: INTEGER;
            F2: CHAR
          END;
CONST K1: T1 = ((F1:1);
                (F2: 'A');
                (F1:2);
                (F2: 'B'));

TYPE T2 = RECORD
     F1: ARRAY [1..2]
               OF INTEGER;
     F2: ARRAY [1..3]
                   OF CHAR
     END;
CONST K2: T2 = (F1: (1, 2);
                F2: ('A', 'B', 'C'));

TYPE T3 = ARRAY [1..2]
                OF SET OF 0..7;
CONST K3: T3 = ([0, 1], [6, 7]);
```

Für Arrays mit dem Komponententyp CHAR gibt es zwei mögliche Formen:

```
TYPE T = ARRAY [1 .. 3] OF CHAR;
CONST K1 : T = ('A', 'B', 'C');
      K2 : T = ('ABC');
```

Wichtig bei der Verwendung typisierter Konstanten ist, daß allen Komponenten eines Arrays oder allen Feldern eines Records Anfangswerte zugewiesen werden müssen.
Typisierte Konstanten dürfen wie Variablen benutzt werden, das heißt, abweichend von der Konzeption einer Konstante darf ihnen während der Programmabarbeitung ein Wert zugewiesen werden.

11.4. Datentypen BYTE und STRING

TURBO-PASCAL stellt zusätzlich die Datentypen BYTE und STRING bereit.
BYTE ist der *Teilbereich* 0 .. 255 von INTEGER.
STRING ist ein Datentyp, dessen Wertebereich *Zeichenketten* einer bestimmten Länge sind, die bei der Deklaration angegeben werden muß. Die maximale Länge ist 255.
Variablen vom Stringtyp dürfen einander zugewiesen und verkettet werden. Als Verkettungsoperator fungiert das Pluszeichen.
Weiterhin ist die Anwendung der Vergleichsoperatoren zulässig.
Beispiele für Operationen mit Zeichenketten sind:

```
VAR S1, S2 : STRING [20];
...
S1 : = 'ABC'
S2 : = S1 + '.TXT';
IF S2 <> 'STOP' THEN ...
IF S1 < S2 THEN ...
```

Ein einzelnes Zeichen einer Zeichenkette kann durch Angabe seines Index ausgewählt werden. Die Zählung beginnt bei 1. Falls S1 z. B. den Wert 'ABC' hat, liefert S1 [3] das Zeichen 'C'.
Weiterhin dürfen Variablen vom Typ STRING als aktuelle Parameter der Standardprozeduren READ, WRITE und ASSIGN angegeben werden.
Das nachfolgende Beispiel zeigt dies:

```
VAR FileName: STRING [10];
    F: FILE OF ... ;
.
.
READ (FileName);
ASSIGN (F, FileName);
.
.
```

Mit READ (FileName) wird der Name des durch ASSIGN zuzuweisenden Files erst zur Programmlaufzeit eingelesen.
Bei der internen Darstellung des Stringtyps wird die maximale Länge mitgeführt. Der Stringtyp STRING [Max] entspricht intern folgendem Rekord:

```
RECORD
  Laenge: BYTE;
  Info: ARRAY [1.. Max] OF CHAR
END
```

Falls die aktuelle Länge kleiner als Max ist, wird die Information durch eine Null beendet. Zur Arbeit mit Stringvariablen gibt es Standardprozeduren und -funktionen. In der nachfolgenden Aufzählung sind s, s1, s2, ... Stringvariablen.

LENGTH (s)	Liefert die aktuelle Länge von s.
POS (s1, s2)	Sucht s1 und s2 und liefert die Position.
COPY (s, pos, n)	Liefert Stringvariable, die von pos beginnend aus n Zeichen von s gebildet wird.
CONCAT (s1, s2, ...)	Liefert Stringvariable aus Kettung von s1, s2, ..
DELETE (s, pos, n)	Entfernt aus s ab pos n Zeichen.
INSERT (s1, s2, pos)	Fügt s1 ab pos in s2 ein.
STR (ausdruck, s)	Konvertiert ausdruck in Stringvariable s.
VAL (s, variable, pos)	Konvertiert s in die interne Darstellung von variable. Enthält s ein nicht erlaubtes Zeichen, zeigt pos seine Position an.

11.5. Absolute Variablen

Mit absoluten Variablen kann die Speicherplatzordnungsstrategie des Compilers umgangen und einer Variablen eine *vorgegebene Speicheradresse* zugeordnet werden. Die folgenden Beispiele zeigen die Deklaration des Bildwiederholspeichers als zweidimensionales Array von CHAR und den Zugriff auf die Kommandozeile des Betriebssystems:

```
VAR BWS : ARRAY [1 .. 24, 1 .. 80]
          OF CHAR ABSOLUTE ○F800;
    .
    .
    KomZeile: STRING [127]
              ABSOLUTE ○80;
    .
    .
WRITELN ('KomZeile: ',Cmd)
```

(Der Zugriff auf die Kommandozeile funktioniert nur, falls das PASCAL-Programm als COM-File durch das Betriebssystem gestartet wurde).
Eine weitere Möglichkeit, in die Speicherplatzzuordnungsstrategie des Compilers einzugreifen, ist das „Übereinanderlegen" von Variablen.

```
VAR XREAL : REAL
    XArr : ARRAY [1 .. 6] OF BYTE
           ABSOLUTE XReal;
```

Die Variablen XReal und XArr haben die gleiche Position im Hauptspeicher.

11.6. Vordeklarierte Arrays

TURBO-PASCAL stellt die vordeklarierten Arrays

```
VAR MEM : ARRAY [0 .. ○FFFF]
          OF BYTE;
    PORT : ARRAY [0 .. ○FF]
          OF BYTE;
```

bereit. MEM korrespondiert mit dem *Hauptspeicher* und PORT mit den *E/A-Toren* des Rechners.
Die Anweisungen
PORT [n]: = ...;
... : = PORT [n];
werden in die Maschinenbefehle
OUT n und
IN n
umgesetzt.

11.7. Einfügen von Maschinenkodepassagen

Maschinenkodepassagen können mit der Inline-Anweisung wie eine gewöhnliche Anweisung in ein PASCAL-Programm eingefügt werden. Die Maschinenbefehle sind in ihrer numerischen Darstellung anzugeben.
Zur Erleichterung dürfen jedoch Variablenbezeichner verwendet werden. Der Compiler ersetzt sie durch ihre Adressen. In TURBO-PASCAL ist das problemlos möglich, da allen Variablen eine feste Hauptspeicheradresse zugeordnet ist (siehe Pkt. 5.5.4.). Eine Bezugnahme auf den Speicherplatzzuweisungszähler ist mit Hilfe des Zeichens "*" möglich.
Das folgende Programmstück zeigt den Befehl LD HL, (X):

```
VAR X: INTEGER
...
INLINE (○2A/X);
...
```

Die einzelnen Bytes bzw. Variablenbezeichner sind durch Schrägstriche zu trennen.

11.8. Aufruf von Maschinenkodeunterprogrammen

TURBO-PASCAL gestattet den Aufruf von Maschinenkodeunterprogrammen. Dazu ist wie bei einer PASCAL-Prozedur die Angabe eines Prozedurkopfes notwendig. Der Block wird jedoch durch den reservierten Bezeichner EXTERNAL und die Adresse des Maschinenkodeprogramms ersetzt.
Nachfolgend ein Beispiel für eine Prozedur mit zwei Parametern:

```
PROCEDURE MaschKode
    (VAR A: INTEGER; B: REAL);
        EXTERNAL adresse;
```

Aufgerufen wird das Maschinenkodeunterprogramm wie eine PASCAL-Prozedur.
Vom Programmierer ist abzusichern, daß zur Abarbeitungszeit an der angegebenen Speicheradresse tatsächlich das erwartete Programm steht.

wird fortgesetzt

Elektronische Baugruppen und mechanische Aufbausysteme für die Automatisierungstechnik

Lothar Kampe, Friedrich Kowarsch
VEB Elektro-Apparatewerke Berlin-Treptow „Friedrich Ebert“, Zentrum für Forschung und Technologie

Modularer Aufbau von Mikroprozessorsystemen

Der Einzug von Mikroprozessoren in die Rechen- und Automatisierungstechnik führte international zum Einsatz von modularen Systemen der elektronischen Baugruppen und der Gefäßtechnik.

In den letzten Jahren hat sich die 19-Zoll-Gefäßtechnik entsprechend den Publikationen der Internationalen Elektronischen Kommission IEC, weltweit durchgesetzt. Die Vereinheitlichung der Maße führte im NSW zu gleichen Aufbauten der Gefäßeinheiten für die unterschiedlichsten Anwendungsfälle in den Industriezweigen. Im RGW wurden die Systemmaße im ST RGW 834–77 und ST RGW 3266–81 standardisiert.

Die Überarbeitung beider Standards erfolgt mit Ziel, eine vollständige Übereinstimmung zu den zwischenzeitlich verbindlichen IEC-Publikationen 297 zu erreichen. Da die DDR seit 1982 auch Mitglied in der IEC ist, werden diese Standards ebenfalls von der DDR mitgetragen.

Die Orientierung auf modulare Systeme, die von internationalen Firmen komplett angeboten werden, reichen von Kunststoffgehäusen für kleine Meßgeräte bis hin zu komplexen und vielseitig verwendbaren Gefäßaufbauten für Steuerungs- und Überwachungseinrichtungen. Dies bedeutet für die Anwender Einsparung an Konstruktions- und Entwicklungskapazität und Einsatz von erprobter und bewährter Gefäßtechnik. Entsprechende positive Erfahrungen und ökonomische Einschätzungen liegen für das ca. 1972 in der DDR entstandene einheitliche Gefäßsystem EGS I/II ebenfalls vor. Durch den modularen Aufbau ist eine Aufrüstung zusätzlicher elektronischer Baugruppen mit modernen Schaltkreisen, um die Anwendungsmöglichkeiten der Einrichtungen zu erweitern, leicht möglich. Die Gefäßsysteme bestimmen durch den Einsatz neuer Technologien, hochwertiger Werkstoffe und modernen Designs den Weltstand. Die Einschubeinheiten der unterschiedlichen, international führenden Firmen sind in ihrem konstruktiven Aufbau nahezu gleich und maßlich kompatibel. Nur die äußeren Hüllen, also die Gehäuse, sind unterschiedlich ausgeführt. Es zeichnet sich der Trend ab, daß die sogenannte Industrieelektronik immer mehr in gutgestaltete Büromöbel Einzug hält. Unter Nutzung der 19-Zoll-Einschubtechnik sind in den letzten 10 Jahren eine Vielzahl von Bussystemen entwickelt worden, die sehr viel detailliertes Wissen aus unterschiedlichsten Bereichen, z. B. der Rechnerarchitektur, Elektronik, Fertigungstechnik und Kenntnisse der Abschirmprobleme erfordern. Diese Bussysteme werden für die 8, 16- und 32-Bit-Mikroprozessortechnik mechanisch passend zum 19-Zoll-Gefäßsystem angeboten.

Gedruckte Rückverdrahtung (GRV)

Die Entwicklung von 16-Bit-Mikroprozessorsystemen erfordert heute eine verstärkte Konzentration auf das Problem des verwendeten Busses. Die Erfahrungen bei 8-Bit-Prozessoren haben gezeigt, wie wichtig für den Erfolg einer Baugruppen-Serie die Entscheidung für einen weitverbreiteten und standardisierten Bus ist, um Parallelentwicklungen und -produktionen zu vermeiden. Wenn man beim Entwurf eines Systems von der funktionellen Struktur einmal absieht, müssen zuerst die elektrischen Voraussetzungen und die mechanischen Paßfähigkeiten erfüllt sein, die meistens als zweitrangig eingestuft werden, aber aus praktischen Erfahrungen ausschlaggebend sind. Die Einschubtechnik stellt die eigentliche Verbindung zwischen der eingesetzten Bauelementegeneration und dem Bus her. Deshalb bieten bekannte Gefäßfirmen bereits konfektionierte gedruckte Rückverdrahtungsleiterplatten sowie auch Stromversorgungen an und überschreiten damit die Grenze zwischen Mechanik und Elektronik.
Zur Paßfähigkeit wurden die mechanischen Anschlußparameter für Bussysteme nach DIN 41494/05 in der IEC im TC 48D – „Elektromechanisches Zubehör für elektronische Geräte“ sowie auch im SC 47B – „Mikroprozessoren“ standardisiert.

Wahl des Kartenformates

Das Europakartenformat mit den Abmessungen (3U) $100 \times 160\,mm^2$ und (6U) $233{,}35 \times 160\,mm^2$ hat sich nicht nur in Europa durchgesetzt, sondern findet auf dem amerikanischen Markt wachsendes Interesse. Folgende Gründe sind für die Durchsetzung der Zollabmessungen für die Karten ausschlaggebend gewesen:

- Die Industrienormen DIN und IEC für Kartenformate, Steckverbinder und Einschubtechnik existierten bereits.
- Dadurch ist ein großes Angebot an mechanischem Zubehör wie Gehäusen, Einschubeinheiten, Steckverbindern usw. vorhanden, die erprobt sind, sich bewährt haben und lieferbar sind.
- Die indirekten Steckverbinder sind wesentlich zuverlässiger als die vorwiegend in den USA eingesetzten direkten Steckverbinder.
- Das kompakte Europakartenformat bildet den besten Kompromiß zwischen Leistungsfähigkeit und Modularität moderner, auf hochintegrierten Bausteinen basierender Computersysteme.

Der Weg zur Standardisierung der Bussysteme

Für die 8- und 16-Bit-Mini- und Mikroprozessor-Computersysteme wurden 1976 von der Firma Intel auf der Grundlage des Mikroprozessors 8086 Busarchitekturen erarbeitet und zwischenzeitlich als Mulitbus I in der IEEE (USA) und IEC standardisiert.
Daneben wurden für 16-Bit-Computersysteme auf der Grundlage des MC 68000 der Firma Motorola unter Zusammenarbeit internationaler Firmen der VME-Bus (Versa Module Europe) in Europa als Standardbus eingeführt. Die unterschiedlichen Mikroprozessoren, wie z. B. auch Z 8000 von der Firma Zilog, sind schaltungstechnisch mehr oder weniger aufwendig auch an Busstrukturen anderer Bauelementehersteller anpaßbar.
Von der Firma Siemens wurden u. a. die AMS-M bzw. AMS-R 16-Bit-Spezifikationen als europäische Version auf der Grundlage des 8086 entwickelt und angepaßt.
Inzwischen wurden die optimalen Strukturen für den Multibus I in der IEC 47B normiert. Grundlage für die mechanische Paßfähigkeit an die 19-Zoll-Einschubtechnik sind die

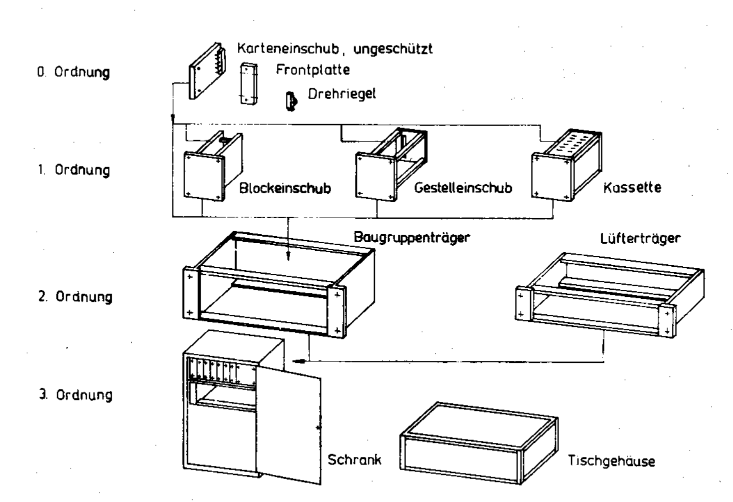


Bild 1 Prinzipieller Aufbau der 19-Zoll-Einschubeinheiten

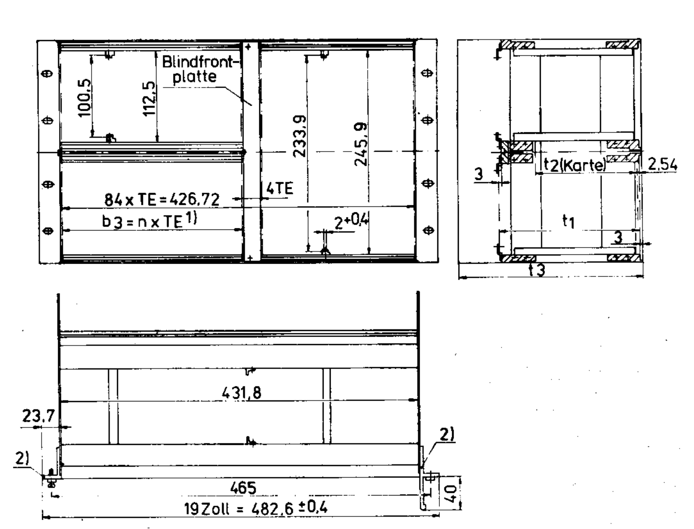


Bild 2 Auszug aus der IEC 279-3

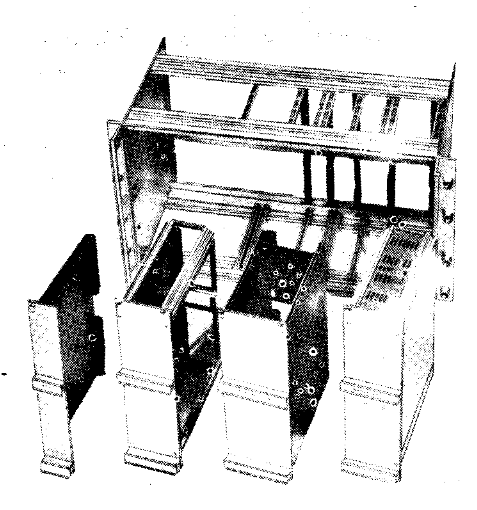


Bild 3 Aufbau- und Bestückungsbeispiel

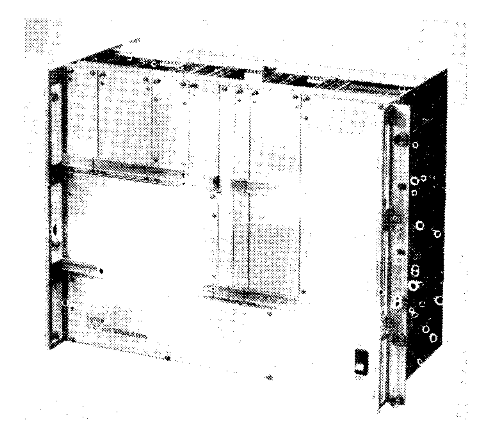


Bild 4 6U-Baugruppenträger auf einem 2U-Lüfterträger

IEC 297–1, 297–2, 297–3 und 3A
Im Multibus-I-System sind die Leiterplattenformate mit 100 × 160 mm^2 (einfachhohe Europakarte) und 233,35 × 160 mm^2 (doppelthohe Europakarte) festgelegt. Die Leiterplatten 100 × 160 mm^2 sind grundsätzlich mit Steckverbindern Bauform C 96polig, nach DIN 41612 oder IEC 603-2 (E TGL 43887) bestückt. Sofern Ein- und Ausgangssignale zu kontaktieren sind, müssen dafür frontseitig Steckverbinder verwendet werden, deren Ausführung dem einzelnen Hersteller überlassen ist (z. B. Interface-Steckverbinder aus Gornsdorf 9/15/25/polig, EBS-GO 4006).
Für zukünftige Mikroprozessorsysteme mit 32 Bit Verarbeitungsbreite wurde 1982 von der Firma Intel der Multibus II zuerst spezifiziert und stellt eine Weiterentwicklung zum Multibus I dar. Entsprechend dem Multibus II wurde in Europa der OSM-B 16- und 32-Bit-Bus für Mikroprozessorsysteme mit einem kompletten Funktionsspektrum spezifiziert (OMS-B = offenes Siemens Mikrocomputer-Baugruppensystem).
Das Format der Multibus-II-Leiterplatten ist wiederum paßfähig zur IEC 297–2, 297–3 und 3A. Auf Grund des günstigen Preis/Leistungsverhältnisses wurde auf die doppelte Europakarte mit der 220-mm-Tiefe orientiert. Es werden ebenfalls die 96poligen Steckverbinder nach DIN 41612 oder IEC 603–2 eingesetzt.
Der Multibus II befindet sich im SC 47B der IEC in der Phase der Standardisierung.

Einschubeinheiten 19 Zoll des KEAW

Zur Aufnahme von elektronischen Baugruppen werden im KEAW 19 Zoll-Einschubeinheiten entwickelt und als kombinatsinterne Lösung in die Produktion überführt. Dieses Gefäßsystem wird das Aussehen der EAW-electronic für Steuerungs- und Überwachungssysteme mit 16 und 32-Bit-Mikrorechnersystemen prägen. Grundlage der Entwicklungen bilden ebenfalls die IEC 297–1, 297–2, 297–3 und 3A.
Damit ist die Paßfähigkeit beim Einsatz von Steckverbindern nach DIN 41612 sowie die Montage der gedruckten Rückverdrahtung eines normierten Bussystems gegeben.
Durch den Einsatz dieser hochentwickelten Aufbautechnik ist es möglich, Mikroprozessorgeräte zu projektieren und die mechanische Anpassung an den eingesetzten Prozessor zu garantieren.
Laufwerkskassetten zur Aufnahme von Floppy-Disk oder Winchester lassen sich bei Bedarf problemlos integrieren. Lüfterbausteine sorgen für die Abfuhr der Wärme.
Weiterhin wurde eine Lösung zum Einbau eines 12"-Bildschirmes im 6U-hohen Baugruppenträger erarbeitet.
Durch den Einsatz der gedruckten Rückverdrahtung (Busleiterplatte) mit mehreren Steckplätzen kann das System bei Erfordernis erweitert werden.
Eine metallische Verkleidung des Baugruppenträgers dient zur Einhaltung der erforderlichen Sicherheitsbestimmungen sowie den Vorschriften zur Funkentstörung und dem Schutz der Signale vor elektromagnetischer Beeinflussung.
Nichtbenutzte Steckplätze werden durch Blindplatten verdeckt.

Entwicklungsergebnis

Bild 1 stellt den prinzipiellen Aufbau der Einschubeinheiten 19 Zoll und die Verknüpfung einiger Gefäße als Beispiel dar, ohne vollständig zu sein.
Das Grundraster ist 2,54 mm.
Folgende Teilungseinheiten sind abzuleiten:
– Teilungseinheit der Breite: TE = 5,08 (n × TE)
– Teilungseinheit der Höhe: U = 44,458 mm (n × U) = Höheneinheiten; h_1
– Teilungseinheit der Höhe für die Leiterkarte:
(3 × U) −35,35 = z. B. 3Höheneinheiten; h_2 = 100 mm
– Einheit des Tiefensprungs: t = 60 mm (t_1 = 175,24 + n x 60)
Zählung n = 1, 2, 3...
Bild 2 stellt einen Auszug aus der IEC 297–3 dar und beinhaltet nur die Hauptmaße im Baugruppenträger.
Im Bild 3 ist ein Aufbau- und Bestückungsbeispiel eines Baugruppenträgers mit Gefäßen 0. und 1. Ordnung dargestellt.
Bild 4 stellt einen 6U-Baugruppenträger auf einem 2U-Lüfterträger dar.

Konstruktive Gestaltung

Das konstruktive Konzept für die Einschubeinheiten ist gekennzeichnet durch eine bestimmte Anzahl beliebig miteinander kombinierbarer Einzelteile, die mittels Schraubverbindung gefügt werden.
Der Grundaufbau für den Baugruppenträger besteht aus 4 Trägerschienen aus Al-Strangpreßprofilen, die mit Stahlseitenwänden und je nach Anwendungsfall erforderlichen Griff- oder Winkelprofilen vom Anwender verschraubt werden. Die Trägerschienen sind zur Aufnahme von Mutternkäfigen oder Gewindestreifen mit entsprechenden Längsnuten ausgebildet. Zur Platzbestimmung der Führungsschienen und auch der Steckverbinder sind selbstklebende Bezeichnungsstreifen in die Trägerschienen einsetzbar. Die Platzkennzeichnung im Baugruppenträger für die Baugruppen können dabei im eingeschobenen Zustand durch eine Bohrung in der Frontplatte abgelesen werden.
Die Führungsschienen sind im Raster n × 5,08 mm einrastbar. Der 6U-hohe Baugruppenträger kann jederzeit und ohne mechanische Änderung mit zusätzlichen Teilen zu einem 3U- und 6U-Mischbaugruppenträger aufgerüstet werden.
An der Verdrahtungsebene ist die Montage der gedruckten Rückverdrahtung nach IEC 47B oder auch bei herkömmlicher Wickelverdrahtung die Montage von Strom- und Kammschienen möglich.
Bei Bedarf können in die Trägerschienen Abdeckbleche oben und unten eingesetzt und auch die rückwärtige Verdrahtungsebene mit einer Abdeckhaube versehen werden.
Die Frontplatten an den Baugruppen (Karteneinschübe, Blockeinschübe, Gestelleinschübe und Kassetten) bestehen aus 2,5 mm Al-Blech, die mit Griffleisten zum Ziehen und mit M 2,5 Befestigungsschrauben ausgerüstet sind. Die Griffleisten sind so ausgebildet, daß Bezeichnungsstreifen eingelegt werden können. Karteneinschübe ohne Frontplatten sind mit Drehriegeln versehen.
Die Leiterkarten sind im Bedarfsfall, z. B. bei Einsatz von SMD-Bauelementen, um 5,08 mm eingerückt aus der Systemlage an der Frontplatte montierbar. Die Leiterkarten sind mit Steckverbindern der Bauform C96, R96 und/oder F48 nach DIN 41612 auszurüsten.
Als Prozeßsteckverbinder werden zusätzlich Interface-Steckverbinder nach IEC 99 eingesetzt (entspricht EBS-GO 4006/01 des VEB KSG).
Da die Steckverbinder am Baugruppenträger fest verschraubt oder an der *Gedruckten Rückverdrahtung* eingelötet sind, besitzen die Plastschienen zur Führung der Leiterkarten zum Toleranzausgleich verbreiterte Führungsnuten.
Die Lüfterträger, die mit den gleichen Al-Strangpreßprofilen wie die Baugruppenträger aufgebaut werden, sind 2U hoch und können mit maximal 3 Lüftern (1459.1, 2, 3 TGL 36958) ausgerüstet werden.

✉ KONTAKT ☏

VEB EAW Berlin, ZFT, Storkower Straße 101, Berlin, 1055; Tel. 4388587 (Kampe) Foto: Eckelt (1)

TERMIN

15. Leipziger Automatisierungskolloquium „CAD-Systeme auprocad-16 für die Projektierung von Automatisierungsanlagen"
WER? AG(B) Automatisierungstechnik-CAD/CAM beim Bezirksvorstand Leipzig der KDT
Wann? 19. Oktober 1988
WO? Leipzig
WAS? Rechnergestützte Projektierung von Automatisierungsanlagen; Softwareproduktionsumgebung SPAS zum Entwurf von Binärsteuerungen; Dialog-Auswahl von Bauteilen für Automatisierungsanlagen aus 16-Bit-PC.
WIE? Interessenten wenden sich formlos schriftlich oder telefonisch an KDT, BV Leipzig, PF 40, Leipzig, 7010

Schwendler

Rekursion – eine faszinierende Beschreibungsmöglichkeit

Dr. Horst Birnstiel
Friedrich-Schiller-Universität Jena, Sektion Technologie

Rekursion ist die Beschreibungstechnik, in der etwas durch sich selbst mit veränderten Parametern und einer Abbruchbedingung erklärt wird. Rekursion ist in prozeduralen Sprachen (z. B. PASCAL) für die Formulierung von Prozeduren eine Möglichkeit unter anderen; in deklarativen Sprachen (z. B. PROLOG) ist ihre Bedeutung wesentlich größer. Was IF, FOR, WHILE und REPEAT für PASCAL sind, das sind die Rekursion und das Backtracking für PROLOG. In prozeduralen Sprachen schreibt ein Programm vor, was der Reihe nach, unter welchen Bedingungen, mit welchen Wiederholungen zu machen ist. In deklarativen Sprachen wird das Problem vom Programmierer mit logischen Aussagen (Prädikaten) beschrieben, und der Rechner bestimmt, was er zu tun hat. Expertensysteme werden heute meist mit deklarativen Werkzeugen entworfen. Die kommende 5. Rechnergeneration mit der möglichen Wissensverarbeitung wird neben LISP Prolog als zentrale Sprache verwenden. Probleme rekursiv zu notieren, ist bis zu einer gewissen Angewöhnungsschwelle schwierig und kippt dann in die Selbstverständlichkeit. Man frage Kollegen, die von PASCAL (oder gar von BASIC) kommen und sich PROLOG angeeignet haben: Sie alle hatten (mehr oder weniger) ihre Schwierigkeiten und verstehen inzwischen nicht mehr, daß sie Schwierigkeiten hatten ... In der Informatik-Ausbildung aller Stufen sollte mehr auf die Rekursion eingegangen werden; die Lernschwelle zum Verstehen rekursiver Algorithmen muß frühzeitig abgebaut werden. Nachfolgend sind mit dieser Zielstellung Beispiele für rekursive Notationen zusammengestellt und jeweils in *Mensch-Notation* und in *PROLOG-Notation* angegeben. Kenntnisse in PROLOG sind für das Verstehen nur wie folgt erforderlich:

PROLOG verlangt **Prädikate,** d. h. logische Aussagen. Ein Prädikat hat einen Namen, in runden Klammern, gefolgt von **Argumenten,** die durch Komma getrennt sind. Als Beispiel ist fak(N,F) die Aussage:

Die Fakultät von N ist F. In Prolog programmieren, heißt **Klauseln** (Fakten und Regeln) aufzuschreiben. Die Klausel „Etwas gilt" heißt **Fakt.** Zum Beispiel ist fak(0,1) der Fakt, daß die Fakultät von 0 gleich 1 ist. Allgemein ist die Fakultät von N gleich F, wenn die Fakultät von N–1 gleich F1 ist und F=N∗F1 gilt. Eine solche Klausel „Etwas gilt, wenn das und jenes und ... gilt" heißt Regel. Damit ist für PROLOG die Fakultätsberechnung angebbar als:

```
clauses fak(0,1).
        fak(N,F) if N > 0 and N1=N-1
          and fak(N1,F1) and F=N*F1.
```

Klauseln stehen nach dem Eröffnungswort „clauses", sie enden mit einem Punkt. Statt „if" kann „:–" geschrieben werden, statt „and" ein Komma „,".

Die Fakultät ist durch 2 Klauseln beschrieben. PROLOG nimmt immer zuerst die erste. Es werde z. B. verlangt

„goal fak(3,X), write(X)."

Goal (Ziel) ist gewissermaßen das Hauptprogramm, das auf die Klauseln zurückgreift. „Groß X" ist eine Variable (Variablen beginnen mit einem großen Buchstaben oder einem Unterstrich). PROLOG prüft die erste Klausel fak(0,1). Die paßt nicht auf das Subgoal fak(3,X). Die 2. Klausel verlangt N > 0 (ja, erfüllt), N1=N–1 (ja, N1 wird an 2 gebunden), fak(N1,F1), PROLOG geht wieder zur ersten Klausel, usw. Die ganze logische Beschreibung wird über den Stack (Kellerspeicher) aufgelöst. Zuletzt wird X an 6 gebunden und angezeigt. Write(X) ist ein Standardprädikat zur Ausgabe. Die *Erlösung* aus der Rekursion erfolgt durch die Abbruchbedingung, hier durch den Fakt fak(0,1). Die Abbruchbedingung muß im Programm zuerst stehen, damit sie bei jedem rekursiven Aufruf geprüft wird. fak(3,X) braucht zur Berechnung fak(2,F1), das braucht fak(1,F2), das braucht fak(0,F3) und jetzt sagt der Fakt fak(0,1), daß F3=1 sein muß. Nun ist die Auflösung möglich. All das macht PROLOG selbständig. Der Programmierer muß sein Problem logisch rekursiv beschreiben. Also:

1. Beispiel

Fakultät F einer natürlichen Zahl N berechnen.

Mensch-Notation:

0! = 1 und N! = N∗(N–1)!

PROLOG-Notation: (Das Eröffnungswort „clauses" wird weggelassen)

```
fak(0,1).
fak(N,F) if N > 0, N1=N-1,
fak(N1,F1), F=N*F1.
```

2. Beispiel

Summe der natürlichen Zahlen von 1 bis N

Die rekursive „Mensch-Denkweise" lautet:

(Summe S der Zahlen 1 bis N) =

(Summe S1 der Zahlen 1 bis N1) plus N

mit der Abbruchbedingung „Summe der Zahlen von 1 bis 1 ist 1"

Die *PROLOG-Notation* ist ein *Abschreiben* der Mensch-Notation:

```
summe(1,1).
summe(N,S) if N1=N-1 and
summe(N1,S1) and S=S1+N.
```

3. Beispiel

Druck der natürlichen Zahlen von 1 bis N

Die iterative *Rechner-Notation* liegt auf der Hand:

for i=1 to n do write(i).

Die rekursive *Mensch-Notation* ist fast lächerlich einfach:

(Druck der Zahlen 1 bis N) =

(Druck der Zahlen 1 bis N–1) und Druck von N

Das ist eine typische Rekursion, sehr selbstverständlich und trotzdem muß man sich erst an sie gewöhnen.

Die *PROLOG-Notation* lautet:

```
druck(0).
druck(N) if N1=N-1, druck(N1), write(N).
```

Die Auflösung ist Sache von PROLOG, der Programmierer beschreibt *nur* sein Problem. Übrigens werden mit

druck(0). druck(N) if write(N), N1=N–1, druck(N1).

die Zahlen rückwärts (von N bis 1) ausgegeben.

4. Beispiel

Listen sind in PROLOG rekursive Objekte. Sie werden durch ihre Darstellung als [Listenkopfelement | Restliste] rekursiv bearbeitbar. Das Listenkopfelement ist ein einzelnes Element. Der senkrechte Strich ist das Trennzeichen. Die Restliste ist eine Liste. In der Erklärung einer Liste taucht also eine Restliste auf; die besteht selbst aus Kopf und Rest, usw. So wird Kopfelement nach Kopfelement abgespaltet, schließlich bleibt die leere Liste []. Es gibt Listen von Zahlen, von Strings, von zusammengesetzten Objekten. In *PROLOG*-Programmen wird häufig die Prüfung gebraucht, ob X ein Element der Liste L ist. Das zugehörige Prädikat heißt member(X,L). Die Liste L läßt sich als [Kopfelement | Restliste] darstellen. Und damit gilt die „tolle" rekursive member-Definition:

(X ist Element der Liste L, bestehend aus Kopfelement und Restliste) if (X ist Kopfelement der Liste)

oder (X ist Element der Restliste)

Im ersten Oder-Part interessiert nur der Kopf, die Restliste nicht. Im zweiten Oder-Part interessiert nur die Restliste, der Kopf nicht. Nicht interessierende Variablen werden mit dem Unterstrich (der anonymen Variablen) "_" bezeichnet.

Umgeschrieben lautet die PROLOG-Formulierung:

```
member(X,[X | _]).
member(X,[_ | Rest]) if member(X,Rest).
```

Man könnte die erste Klausel (sie ist oben ein Fakt) auch als Regel schreiben:

member(X,[Kopf | _]) if X=Kopf.

5. Beispiel

Die Länge einer Liste ist die Anzahl ihrer Elemente. Es gibt real-Listen, string-Listen, Listen zusammengesetzter Objekte, Listen von Listen, usw. Die Listen sind in PROLOG und LISP (List-Processing, da kommen die Listen her!) etwas sehr Allgemeines. In prozeduralen Sprachen denkt man bei der Bestimmung der Länge einer Liste sofort ans Abzählen. Nicht so in deklarativer Denkweise:

Die Länge der leeren Liste ist Null.

Die Länge einer Liste ist N, wenn die Länge der Restliste (Liste ohne ihr erstes Element) N–1 ist.

Diese Aussage ist derart selbstverständlich, derart naiv, daß man sich gewissermaßen erst daran gewöhnen muß, zu begreifen, daß sie dem Rechner genügt.

```
listlen([],0).
listlen([_ | Rest],N) if listlen(Rest,N1) and
N=N1+1.
```

6. Beispiel

Ein Prädikat ntesElement(Liste,N,X) soll von einer Liste L = [Kopfelement | Restliste] das N-te Element X bestimmen. Und wieder ist die *logische* Beschreibung der Aufgabe eine Selbstverständlichkeit:

Das an der Stelle 1 stehende Element der Liste L ist das Kopfelement.

Das an der Stelle N stehende Element der Liste L ist X, wenn das an der Stelle N–1 stehende Element der Restliste auch X ist.

In PROLOG-Klauseln lauten diese Aussagen:

```
ntesElement([Kopfelement | _],1,Kopf-
element).
ntesElement([_ | Restliste], N, X) if
N > 1, N1=N-1,
ntesElement(Restliste,N1,X).
```

7. Beispiel
Ein eindrucksvolles Beispiel für die Eleganz rekursiver Problembeschreibung (und für PROLOG auch gleichzeitig Programmierung!) ist für die Formulierung des Spiels „Türme von Hanoi". Es gibt 3 Stapelplätze für Scheiben: links, Mitte, rechts. Links steht ein Turm von N nach oben immer kleiner werdenden Scheiben. Man denke an N=3 mit Geldstücken: Groschen ganz unten, dann Fünfer, dann Pfennig. Der Turm soll von links nach rechts gebracht werden, der mittlere Platz dient während des Spiels als Ablage. Die Spielregeln lauten: 1. Stets nur eine Scheibe bewegen. 2. Stets darf eine Scheibe nur auf einer größeren Scheibe zu liegen kommen. Eine mögliche Lösung für N=3 ist: links nach rechts, links nach Mitte, rechts nach Mitte, links nach rechts, Mitte nach links, Mitte nach rechts, links nach rechts.
Rekursiv läßt sich das Problem allgemein für N Scheiben wie folgt formulieren:
N Scheiben von links unter Nutzung von Mitte nach rechts bringen heißt:
N−1 Scheiben von links unter Nutzung von rechts nach Mitte bringen
UND eine Scheibe von links nach rechts bringen
UND N−1 Scheiben von Mitte unter Nutzung von links nach rechts bringen.
In Prolog lautet das (für ein lauffähiges Programm fehlt noch einiges):

```
bringe(N,Links,Mitte,Rechts) if N1=N−1,
  bringe(N1,Links,Rechts,Mitte),
  write(„von",Links,„nach",Rechts),
bringe(N1,Mitte,Links,Rechts).
```

Links, Mitte, Rechts sind Variablen, denen allen dreien die Konstanten links, rechts, mitte (kleingeschrieben!) zugewiesen werden können.

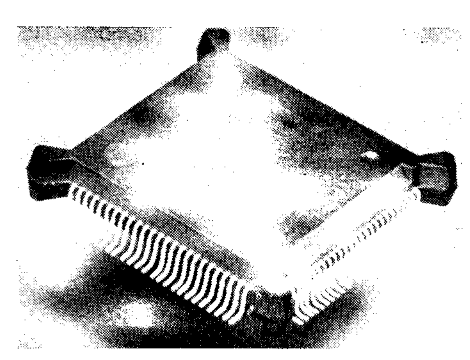
8. Beispiel
Die Rekursion ist grundsätzlich anwendbar, wenn irgendein Problem für N=1 lösbar ist und für N=allgemein angegeben werden soll. Dazu zählen u. a. Fahrtrouten (N=1: von einem Ort zum anderen direkt; N=allgemein: von einem Ort über andere Orte zu irgendeinem), Pläne (z. B. Arbeitspläne) (N=1: eine einzige Aktivität, z. B. ein Arbeitsgang; N=allgemein: alles übrige ohne die erste Aktivität), usw. Als Fakten seien viele Wege von jeweils einem Zustand zu einem anderen samt zugehörigen Übergangsoperationen gespeichert. Bei Wegeproblemen z. B. weg(ort1,ort2,entfernung), bei Planproblemen z. B. teil(zustand1,zustand2,operation), usw.
Dann gilt (wieder selbstverständlich):
„Von Anfang nach Ende kommen" =
„Es gibt einen (direkten) Weg von Anfang nach Ende."
ODER „Es gibt einen Weg von Anfang nach X."
UND „Von X nach Ende kommen."
Das ist schon (fast!) die PROLOG-Programmierung.
PROLOG (= Programming in Logic) unterstützt von den höheren Programmiersprachen am besten Rekursion (davon war hier die Rede) und Backtracking (davon war hier nicht die Rede).
Die angegebenen Beispiele sind in TURBO-PROLOG notiert; andere Dialekte haben geringfügige Abweichungen.
Wir müssen mehr die rekursive Beschreibung von Problemen nutzen, sicher mehr in der Ausbildung, aber wohl auch im Programmieralltag.

⊠ KONTAKT ☏

FSU Jena, Sektion Technologie, Ernst-Thälmann-Ring 32, Jena, 6900; Tel. 8222741

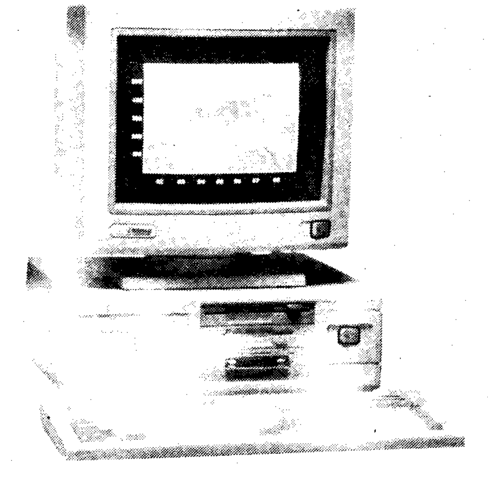
Technik international

32 Bit für Einsteiger

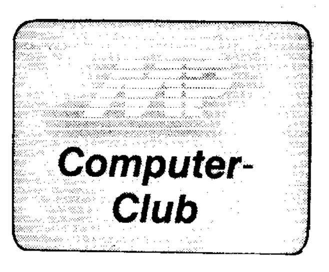


Mit dem Deskpro 386s will Compaq einem großen Anwenderkreis im PC-Sektor die 32-Bit-Technik erschwinglich machen. Basis dafür ist der Einsatz einer „Billig-Version" des Prozessors 80386, bei welcher der externe Datenbus auf 16 Bit reduziert wurde – immer noch ausreichend für eine Leistung von 2,5 bis 3 MIPS bei einer Taktfrequenz von 16 MHz.

Werkfotos Compaq



Seit kurzem produziert Intel eine abgerüstete Variante des 32-Bit-Prozessors 80386 unter der Bezeichnung 80386SX (früher P9), die im Gegensatz zum „echten" 32-Bit-80386 extern nur mit 16 Bit Datenbreite arbeitet. Das spart nicht nur Konstruktions- und Herstellungsaufwand, sondern reduziert auch die Kosten für ein Rechnersystem, denn statt einer 32-Bit-Peripherie wird nur solche mit 16 Bit benötigt. Ähnliches praktizierte Intel ja bereits beim 8088, der gegenüber dem „echten" 16-Bit-Prozessor 8086 extern nur mit 8 Bit arbeitet; IBM nutzte dies dann für die Massenproduktion ihrer ersten PCs aus, da der 8088 billiger und schneller in großen Stückzahlen verfügbar war.
Der 80386SX ist zum 80386 voll befehlskompatibel, mit seinen 24 externen Adreßleitungen verwaltet er bis zu 16 MByte physischen Arbeitsspeicher. Getaktet wird er mit 16 MHz; damit soll seine Rechenleistung nur etwa 10 Prozent geringer sein als die des 80386.
Compaq nutzt nun in Zusammenarbeit mit Intel den 80386SX, um den „ersten Personal Computer mit 80386-Prozessor für breite Anwenderkreise" zu produzieren – den Compaq Deskpro 386s – und damit seine Produktpalette nach unten abzurunden. Der 386s ist als Einstieg in die Familie der 80386-Rechner beispielsweise bei der Bürokommunikation mit Standardanwendungen oder als Basis-Workstation unter MS OS/2 gedacht. Lauffähig ist die bereits vorhandene Software unter MS-DOS, MS Windows/386 und OS/2. Andererseits kann die auf dem 386s entwikkelte Software später auch auf „echten" 32-Bit-PCs weitergenutzt werden. Gegenüber 80286-Systemen soll der 386s eine zwischen 35 und 60 Prozent höhere Systemleistung besitzen.
Durch Einsatz von Enhanced Page Memory, auf den der Prozessor über einen schnellen 16-Bit-Bus zugreift, wird die Leistung des 80386SX-Prozessors im 386s optimiert. Um die Systemleistung bei rechenintensiven Anwendungen wie Tabellenkalkulation weiter zu erhöhen, ist eine Ausstattung mit dem wahlweise verfügbaren Intel-80387SX-Koprozessor möglich.
Der enorme Datendurchsatz bei allen Systemkomponenten wird durch einen getrennt arbeitenden „concurrent"-Bus erreicht, der für Zugriffe auf Hauptspeicher und Peripherie getrennte Buswege zur Verfügung stellt. Damit können alle Systemkomponenten mit der optimalen Geschwindigkeit getaktet werden, ohne an Kompatibilität zum Industriestandard zu verlieren.
Der „concurrent"-Bus verknüpft einen Hauptspeicherbus mit dem Industriestandard-Peripheriebus und arbeitet mit der gleichen Taktrate wie der 16-MHz-80386SX-Prozessor. Dieser ist auf die Leistungsanforderungen von Systemerweiterungen wie Festplattenlaufwerken, Video-Systemen, Netzwerk-Karten, DFÜ-Optionen und ähnlicher Peripherie abgestimmt. Die Integration zahlreicher Systemfunkionen in ASICs sowie der intensive Einsatz von Surface-Mount-Technology (SMT) auf der Systemplatine ermöglichen die geringen Ausmaße des 386s. Durch Konzentration verschiedener Baugruppen auf der Systemplatine (u. a. auch der Video-Grafik-(VG-)-Controller, Festplatten- und Standard-Schnittstellen) bleiben vier 8/16-Bit-Steckplätze frei.
Damit ist im Gehäuse Platz für vier Massenspeicher und vier 8/16-Bit-Erweiterungsplatinen mit voller Länge.
Während das Modell 1 des 386s über ein 5,25-Zoll-Floppy-Laufwerk verfügt (optional 3,5 Zoll), besitzen das Modell 20 zusätzlich ein 20-MByte- und das Modell 40 ein 40-MByte-Festplattenlaufwerk. Wahlweise gibt es auch 110-MByte-Festplattenlaufwerke mit weniger als 25 Millisekunden Zugriffszeit. Der 386s unterstützt bis zu zwei Diskettenlaufwerke und bis zu zwei Festplattenlaufwerke. Für letztere gibt es eine Disk-Cache-Software, die den Zugriff beschleunigt und dadurch die Systemleistung erhöht. Ebenfalls als Option gibt es ein 135-MByte-Bandlaufwerk. Alle Modelle sind mit 1-MByte-Enhanced-Page-RAM ausgestattet. Der Systemspeicher läßt sich modular bis auf 13 MByte schnellen RAM erweitern. Im Lieferumfang ist auch der COMPAQ Expanded Memory Manager CEMM enthalten. Diese Software ist kompatibel zu der Lotus/Intel/Microsoft LIM/EMS-Spezifikation und ermöglicht, die 640-KByte-Grenze unter MS-DOS zu durchbrechen und auf den erweiterten Speicher zuzugreifen. *MP*



V 24-Treiberroutine für den Plotter K 6418

Klaus-Dieter Kirves; Karsten Schiwon
VEB Mikoroelektronik „Wilhelm Pieck" Mühlhausen

Die Anschlußmöglichkeit von grafikfähigen Druckern wurde bereits in vorangegangenen Ausgaben beschrieben, hier ist nun eine weitere Möglichkeit des Anschlusses eines grafikfähigen Ausgabegerätes, eines Plotters, für die Kleincomputer KC 85 vorgestellt.

Allgemeines

Wesentlicher Unterschied zu den bisherigen Treiberprogrammen ist das verwendete V.24-Software-Protokoll. Dieses Protokoll wird auch DC1-DC3-Protokoll genannt. Wird der Empfang von Daten von der Steuerung des Plotters erlaubt, sendet dieser ein DC1, der Kleincomputer kann Daten an den Plotter ausgeben. Ist der Pufferspeicher des Plotters bis auf 500 Byte belegt, wird vom Plotter das Signal DC3, Puffer voll, gesendet. Prinzipiell ist es möglich, weitere 12 Byte fehlerfrei in den Pufferspeicher des Plotters zu übergeben. Es ist allerdings weniger aufwendig, den Datenfluß sofort zu unterbrechen, so wie das auch der in Bild 1 abgebildete Treiber macht. Das Senden von Daten vom Computer wird erst nach Erkennen des DC1-Signals wieder erlaubt. Wird ein Byte fehlerhaft vom Plotter empfangen, sendet dieser die Meldung durch ein Byte DC4.

Auf diese Weise ist eine Synchronisation des Datenflusses, ähnlich wie über die Steuerleitung beim Hardware-Protokoll möglich. Weiterhin ist es möglich, bei Verarbeitung des empfangenen Fehlersignals DC4, eine Statusabfrage des Plotters vorzunehmen. Dazu muß an den Plotter ein EHQ (05H) ausgegeben werden. Als Antwort sendet der Plotter sein Statusbyte, das, wie im Bild 2 dargestellt, aufgebaut ist.

Zum besseren Verständinis ist im Bild 3 der Datenfluß zum Softwareprotokoll dargestellt.

Diese Protokollart verlangt allerdings, daß beide zusammengeschlossenen Geräte über Sender und Empfänger verfügen. Neben den zwei Datenleitungen wird nur die Masseverbindung benötigt. Der Anschluß des V.24-Kabels erfolgt am Kleincomputer über einen Diodenstecker, am Plotter über eine 26polige Steckerleiste nach TGL 29331, Blatt 04 (Griffelemente nach TGL 29331, Blatt 08). Es sind die in Bild 4 dargestellten Anschlüsse zu verbinden.

Bedienung der Treibersoftware

Der Einheitlichkeit wegen erfolgt die Bedienung der Treibersoftware so, wie das von den Treiberprogrammen der Drucker bekannt ist.

Das Teiberprogramm PLOT-6418COM belegt im KC-85-System den Speicherbereich von BA00H (Anfangsadresse) bis BC00H (Endadresse). Es ist natürlich möglich, dieses Programm auf andere Adressen zu binden, so ist z. B. bei gleichzeitiger Nutzung eines Druckers mit der Ansteuersoftware der Programmmkassette C0/171 V.24-Software das Binden auf die Adresse BC00H zu empfehlen. Die Selbsstartadresse ist BA0BH, so, daß nach dem Einlesen selbständig eine Initialisierung eines V.24-Moduls M003 des KC-85-Systems erfolgt:

- Aktivierung des V.24-Moduls im Modulschacht 8
- Zeichenausgabe über Kanal 1 des V.24-Moduls
- Initialisierung des USER-Ausgabekanals 1 (USER OUT 1, Basic LIST#2).

Durch den Selbststart des Programms werden folgende Übertragungs- Bedingungen, die dem vom Plotter angeforderten Datenformat entsprechen, eingestellt:

Übertragungsrate:	1200 Baud
Bit pro Zeichen	7
Anzahl der Stoppbits:	2 (mindestens 1 verlangt)
Parität:	ungerade

Neben dem Auto-Start ist, wie nach einem RESET des Systems notwendig, ein Start der Treiberroutine über das CAOS-Menü möglich. Durch Anwahl des Menü-Eintrages *PLOT6418* wird, ohne die Angabe von Parametern, die Ausgabe über V.24-Interface für den Plotter erneut initialisiert. Bei der Initialisierung des Programmes kann der Modulschacht angegeben werden, in welchem der Modul M003 kontaktiert ist. Der Aufruf des Programmes erfolgt aus der Menüeingabe des Betriebssystems heraus.

Folgendes Beispiel verdeutlicht dies:

·/·PLOT6418C

Hier wird der Modul im Schacht mit der Adresse 0C initialisiert.

Tafel 1 Initialisierungstabelle für Übertragungsraten zur Ansteuerung des Plotters K 6418; Vollduplex, 7 Bit/Zeichen, 2 Stopp-Bit

Übertragungsrate Bit/s	1	2	3	4	5	6	7	8	9
2400	47	17	18	04	4D	03	41	05	28
1200	47	2E	18	04	4D	03	41	05	28
600	47	5B	18	04	4D	03	41	05	28
300	47	60	18	04	89	03	41	05	28
150	47	5B	18	04	C9	03	41	05	28

(Spaltenüberschrift: Initialisierungsbytes)

***Bild 1** Assemblerlisting des V.24-Treiberprogrammes zum Anschluß eines Plotters K 6418 an das Kleincomputersystem KC 85*

```
3000                 ORG     0BA00H
BA00          ;V.24-TREIBER FUER PLOTTER K6418
BA00          ;  K. SCHIWON & K.-D. KIRVES
BA00          CAOS    EQU     0F003H  ;SYSTEMAUFRUF
BA00          ARGN    EQU     0B781H  ;ANZAHL ARGUMENT
BA00          ARG1    EQU     ARGN+1  ;ARGUMENT 1
BA00          OUTAB   EQU     0B7B9H
BA00          UOUT1   EQU     0B7BDH
BA00          SIO     EQU     08H
BA00          CTC     EQU     0CH
BA00          DC1     EQU     11H     ;PUFFER LEER
BA00          DC3     EQU     13H     ;PUFFER VOLL
BA00          DC4     EQU     14H     ;FEHLERMELDUNG
BA00          EQH     EQU     05H     ;FEHLERANFORDER.
BA00 47       INTAB:  DEFB    47H     ;CTC-START
BA01 2E               DEFB    2EH     ;1200 BAUD
BA02 18               DEFB    18H     ;SIO-RESET
BA03 04               DEFB    4       ;->WR4
BA04 4D               DEFB    4DH
BA05 03               DEFB    3       ;->WR3
BA06 41               DEFB    41H
BA07 05               DEFB    5       ;->WR5
BA08 28               DEFB    28H
BA09          INTABE  EQU     *
BA09 01       HCPZ:   DEFB    1       ;BIT0 - ERROR
BA0A                                  ;BIT7 - PUFFER?
BA0A 08       MODSCH: DEFB    8       ;MODULSCHACHTNR.
BA0B C323BA           JP      START   ;SELBSTSTARTADR.
BA0E 7F7F             DEFW    7F7FH
BA10 5632344B         DEFM    'V24K6418'
BA18 01               DEFB    1
BA19 3A81B7           LD      A,(ARGN)
BA1C A7               AND     A
BA1D 2804             JR      Z,START
BA1F 7D               LD      A,L
BA20 320ABA           LD      (MODSCH),A
BA23 3A0ABA   START:  LD      A,(MODSCH)
BA26 47               LD      B,A
BA27 0E80             LD      C,80H
BA29 ED78             IN      A,(C)
BA2B FEEE             CP      0EEH    ;M003?
BA2D 2805             JR      Z,ST1
BA2F CD03F0   ERROR:  CALL    CAOS
BA32 19               DEFB    19H     ;FEHLERMELDUNG
BA33 C9               RET
BA34 68       ST1:    LD      L,B
BA35 3E02             LD      A,2
BA37 1601             LD      D,1
BA39 5A               LD      E,D
BA3A CD03F0           CALL    CAOS
BA3D 26               DEFB    26H     ;SWITCH
BA3E 2192BA           LD      HL,SDD
BA41 22BEB7           LD      (UOUT1+1),HL
BA44 0E0C             LD      C,CTC
BA46 0602             LD      B,2
BA48 2100BA           LD      HL,INTAB
BA4B EDB3             OTIR
BA4D 0E0A             LD      C,SIO+2
BA4F 0607             LD      B,7
BA51 EDB3             OTIR
BA53 CD03F0           CALL    CAOS
BA56 23               DEFB    23H     ;TEXTAUSGABE
BA57 44727565         DEFM    'Druecke PAPER am K6418'
BA6D 0D0A             DEFW    0A0DH
BA6F 00               DEFB    0
BA70 DB0A     W1:     IN      A,(SIO+2);RR0
BA72 CB47             BIT     0,A     ;DATEN IM PUFFER
BA74 2008             JR      NZ,W2
BA76 3E10             LD      A,10H
BA78 CD03F0           CALL    CAOS
BA7B 14               DEFB    14H     ;WARTESCHLEIFE
BA7C 18F2             JR      W1
BA7E DB08     W2:     IN      A.(SIO) :DATEN
```

```
BA80 E67F              AND     7FH       ;AUSBLENDEN BIT7
BA82 FE11              CP      DC1       ;BEREIT?
BA84 20A9              JR      NZ,ERROR
BA86 CD03F0            CALL    CAOS
BA89 23                DEFB    23H
BA8A 4F2E4B2E          DEFM    'O.K.'
BA8E 0D0A              DEFW    0A0DH
BA90 00                DEFB    0
BA91 C9                RET
BA92            ;ZEICHENAUSGABE
BA92 A7         SOD:   AND     A         ;DUMMY
BA93 C8                RET     Z
BA94 C5                PUSH    BC
BA95 F5                PUSH    AF
BA96 3A09BA            LD      A,(HCPZ)  ;DC3 alt
BA99 CB7F              BIT     7,A
BA9B 2029              JR      NZ,SD1
BA9D DB0A              IN      A,(SIO+2);STEUERBYTE
BA9F CB47              BIT     0,A       ;DATEN EINGEG.?
BAA1 2817              JR      Z,SD4
BAA3 DB08              IN      A,(SIO)   ;LESEN BYTE
BAA5 E67F              AND     7FH       ;AUSBLENDEN BIT7
BAA7 FE13              CP      DC3       ;PUFFER VOLL?
BAA9 200B              JR      NZ,SD2
BAAB 0680              LD      B,80H     ;merken
BAAD 3A09BA     SD6:   LD      A,(HCPZ)
BAB0 E67F              AND     7FH
BAB2 B0                OR      B
BAB3 3209BA            LD      (HCPZ),A
BAB6 FE14       SD2:   CP      DC4       ;Fehler?
BAB8 2826              JR      Z,SD3
BABA F1         SD4:   POP     AF
BABB D308              OUT     (SIO),A   ;Ausgabe Byte
BABD 3E03              LD      A,3       ;=18ms
BABF CD03F0            CALL    0F003H
BAC2 14                DEFB    14H       ;WAIT
BAC3 C1                POP     BC
BAC4 A7                AND     A         ;CY=0
BAC5 C9                RET
BAC6 DB08       SD1:   IN      A,(SIO)   ;LESEN EING.
BAC8 E67F              AND     7FH       ;AUSBLENDEN BIT7
BACA FE14              CP      DC4       ;Fehler ?
BACC 2812              JR      Z,SD3
BACE FE11              CP      DC1
BAD0 280A              JR      Z,SD11    ;WARTEN
BAD2 CD03F0            CALL    CAOS
BAD5 2A                DEFB    2AH       ;BRK?
BAD6 30EE              JR      NC,SD1
BAD8 3EFF       SD13:  LD      A,0FFH    ;ABBRUCH
BADA 1819              JR      SD14
BADC 0600       SD11:  LD      B,0
BADE 18CD              JR      SD6
BAE0 F1         SD3:   POP     AF
BAE1 3E05              LD      A,EQH     ;FEHLERNUMMER?
BAE3 D308              OUT     (SIO),A
BAE5 DB0A       SD7:   IN      A,(SIO+2);CONTROL
BAE7 CB47              BIT     0,A
BAE9 2008              JR      NZ,SD12   ;WARTEN AUF BYTE
BAEB CD03F0            CALL    CAOS
BAEE 2A                DEFB    2AH       ;BRK?
BAEF 30F4              JR      NC,SD7
BAF1 18E5              JR      SD13
BAF3 DB08       SD12:  IN      A,(SIO)   ;FEHLERNUMMER
BAF5 CD03F0     SD14:  CALL    CAOS
BAF8 1C                DEFB    1CH       ;AUSGABE HEX
BAF9 CD2FBA            CALL    ERROR
BAFC C1                POP     BC
BAFD 37                SCF               ;CY=1
BAFE C9                RET
BAFF            ;      E N D E

ERRORS: 0000
```

Steht ein DEVELOPMENT-Modul M027 zur Verfügung, so kann das Listing eingegeben und übersetzt werden. Anderenfalls kann auch eine Eingabe des Maschinencodes mittels des CADOS-Kommandos MODIFY ab Adresse BA00H erfolgen. In beiden Fällen wird das Maschinenprogramm als selbststartendes Programm auf Kassette abgespeichert. Die Kommandoeingabe dafür lautet:
·/·SAVE BA00 BB00 BA02
NAME: PLOT6418COM
Alle weiteren Informationen zum Programm können dem Listing entnommen werden.

Eine Änderung der Übertragungsraten ist durch die Verwendung der in der Tafel 1 dargestellten Initialisierungsbytes ab Adresse BA00 möglich.

Die V.24-Schnittstelle des Plotters

Beim Plotter K 6418 lassen sich die Übertragungsbedingungen mit Schaltern auf der V.24-Schnittstellenkarte des Gerätes einstellen. Dazu muß der Gehäusedeckel geöffnet werden. Die Schalter werden beim Einschalten des Plotters oder beim RESET abgefragt. Aus dem Bild 5 ist

```
Bit7   6   5   4   3   2   1   0
 1     1   1   x   1   1   1   1
                                falsches Kommando
                            falscher Parameter
                        falscher Charakter
                    falsches Fenster
              Puffer aufnahmefähig
          Puffer voll
      Plotter bereit
```

Bild 2 Statusbyte des Plotters

```
Datenleitung       DC1                  DC3   DC1   +12V
vom Plotter        I                    I     I
                   I                    I     I     -12V
      ----------------------------------------------
Datenleitung            IIIII III IIII              II+12V
zum Plotter             IIIII III IIII              II
                        IIIII III IIII              II-12V
      ----------------------------------------------
```

Bild 3 Zeitdiagramm des Duplex-V.24-Protokolls

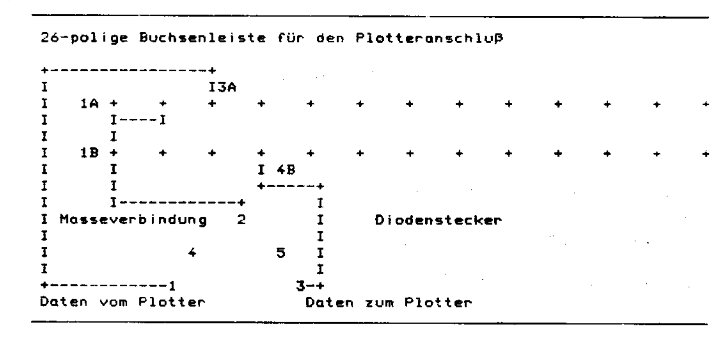


Bild 4 Verbindungsleitung zum Anschluß des Plotters k 6418 an den V.24-Modul des Kleincomputers KC 85/3

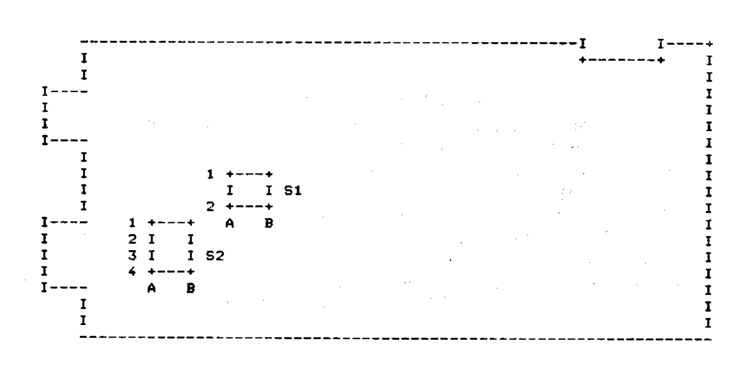


Bild 5 DIL-Schalteranordnung des Plotters-V.24-Bausteins
S1: Schalter zur Interfaceumschaltung (dargestellt V.24)
S2: Geschwindigkeits- und Bitratenumschalter
Schalterstellung: A = on; B = off

Tafel 2 Funktionstabelle S1

S1.1	S1.2	S1.3	S1.4	Bedeutung
on	off	on	off	IFSS
off	on	off	on	V.24

Tafel 3 Funktionstabelle S2

S2.4	S2.3	S2.2	S2.1	Bitrate/Geschwindigkeit
on	on	on		9600 Bit/s
on	on	off		4800 Bit/s
on	off	on		2400 Bit/s
on	off	off		1200 Bit/s
off	on	on		600 Bit/s
off	on	off		300 Bit/s
off	off	on		150 Bit/s
off	off	off		75 Bit/s
			on	240 mm/s
			off	100 mm/s

die DIL-Schalteranordnung des V.24-Bausteins des Plotters ersichtlich. Die Bedeutung der Schalterstellungen sind in Tafel 2 und 3 angegeben. Zusammengefaßt ergeben sich für die Inbetriebnahme des durch den Computer angesteuerten Plotters K 6418 folgende Schritte:
- Einstellen der Datenübertragungsrate am Plotter
- Herstellen des Verbindungskabels
- Kontrolle des Verbindungskabels
- V.24-Modul stecken (dabei darauf achten, daß der Computer ausgeschaltet ist)
- Verbindung zwischen KC und Plotter herstellen
- Einschalten von KC und Plotter
- Laden des Treiberprogramms PLOT6418COM

Der Plotter ist betriebsbereit, wenn die Meldung *Drücke Paper am K6418* erfolgte und die Taste *Paper* am Plotter bedient wurde.
Wurden oben genannte Bedingungen eingehalten, kann die Arbeit mit dem System Computer-Plotter beginnen.

Literatur

/1/ Beschreibung zur C0171 V.24 Software. VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen
/2/ Mikroprozessorsystem der II. Leistungsklasse, Technische Beschreibung, Schaltkreis für Zähler- und Zeitgeberfunktion CTC U 857 D. VEB Mikroelektronik „Karl Marx" Erfurt
/3/ Mikroprozessorsystem der II. Leistungsklasse, Technische Beschreibung, Schaltkreis für serielle Ein- und Ausgabe SIO U 856 D. VEB Mikroelektronik „Karl Marx" Erfurt
/4/ KC 85, Beschreibung zum Modul M003 V 24. VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen
/5/ Betriebsvorschrift K6418.02. Robotron

Bildschirmausdruck in BASIC unter SCP

In MP 3/1988, S. 95, wurde ein BASIC-Programm zum Ausdrucken von Bildschirminhalten beim PC 1715 unter SCP vorgestellt. Da mit dieser Lösung jedes einzelne Zeichen durch eine PEEK-Anweisung aus dem Bildwiederholspeicher gelesen wird, läuft der Druck relativ langsam ab. Es gibt leistungsfähigere Lösungen, diesen Ausdruck von Bildschirminhalten zu realisieren. Neben der Möglichkeit, in Verbindung mit der Installation des SCP entsprechende Vorkehrungen zu treffen (vgl. beispielsweise rd 1/1988, S. 39), kann der Ausdruck – insbesondere wenn er erst in Abhängigkeit von einer bestimmten Bildschirmanzeige entschieden werden soll – auch mit dem nachfolgenden BASIC-Programm erfolgen.

```
1000 'UP zum Ausdruck Bildschirminhalt
1010 '=
1020 X$ = SPACE$ (80)
1030 ADR1 = VARPTR (X$)
1040 ADR2% = -2084
1050 ADR3 = VARPTR(ADR2%)
1060 FOR I% = 1 TO 24
1070 Z1 = PEEK(ADR3)
1080 Z2 = PEEK(ADR3 + 1)
1090 POKE ADR1 + 1, Z1
1100 POKE ADR1 + 2, Z2
1110 LPRINT X$
1120 ADR2% = ADR2% + 80
1130 NEXT I%
1140 RETURN
```

Dieses Programm ist als Unterprogramm angelegt, kann aber auch durch Ersetzen der RETURN-Anweisung durch END als selbständiges Programm aufgerufen (und auch übersetzt) werden. Das Programm ist normalerweise für den Ausdruck von 24 Zeilen des 24 × 80-Bildschirmes vorgesehen, kann aber auch für eine andere Zeilenanzahl bzw. den 16 × 64-Bildschirm ausgelegt werden. Gleichzeitig kann es zweckmäßig sein, das Programm in Abhängigkeit von einer bestimmten Zeichenfolge auf einer Bildschirmzeile (z. B. Ok-Meldung) zu beenden.
Das ist dann durch Einfügen der folgenden Programmzeile möglich:

```
1105 IF LEFT$ (X$, 2) = „Ok" THEN GOTO 1140
```

Als Grundlage für die Arbeit dieses Programms wird die interne Darstellung von Zeichenketten verwendet, wobei jeweils eine Bildschirmzeile der Zeichenkettenvariablen X$ zugeordnet wird.

Dr. Claus Goedecke

KC-85/3 – BASIC – Tip Interruptgesteuerte Echtzeitmessung mit System – CTC

Die Möglichkeiten des BASIC-INTERPRETERS, Zeitmessungen in einem BASIC-Programm zu realisieren, sind sehr uneffektiv. Zum Beispiel ist die reale Erfassung der Zeit zwischen zwei Tastaturbetätigungen nur mit BASIC-Sprachelementen kaum möglich. Für solche Aufgaben ist die Verwendung eines interruptgesteuerten CTC-Kanals immer die günstigste Lösung.
IM KC 85/3 (/2) wird der Kanal 2 des Grundgeräte-CTC zur Realisierung der Blinktaktfrequenz verwendet. Dieser Takt hat eine Periodendauer von etwa 0,8 Sekunden. Der betreffende CTC-Kanal arbeitet als reiner Zähler ohne Interrupt und wird von einem in der internen Hardware erzeugten Takt gespeist. Die Ausgangsimpulse dieses Kanals werden zur Realisierung der Blinkeffekte auf dem Displaybild verwendet. Wird nun in der Interruptstartadressentabelle (01E8H+4) für diesen Kanal die Startadresse für eine kurze Zählroutine eingetragen und die Interruptsteuerung freigegeben, so erhält man einen frei laufenden Zähler und muß vom BASIC-Programm aus nur die entsprechenden Speicherzellen auslesen bzw. durch Überschreiben rücksetzen.
Das folgende Programm realisiert eine solche Zählroutine (ständiges Incrementieren einer Integerzahl auf der Zeile 268–269 bzw. 10CH-10DH) und die entsprechende CTC-Uminitialisierung. Die Zählerroutine wird ab 256–267 bzw. 100H–10BH geladen.

```
10 DATA 251, 229, 42, 12, 1, 35, 34, 12, 1, 225, 237, 77, 0, 0
20 FOR I = 1 TO 14
30 READ J
40 POKE 255 + I, J
50 NEXT I
60 DOKE 492, 256
70 OUT 142, 249
```

Das Auslesen der Zählzelle erfolgt im BASIC-Programm z. B. mit I: = DEEK (268) und das Rücksetzen kann mit DOKE 268,0 realisiert werden. Wird beim Uminitialisieren des CTC-Kanals auch eine neue Zeitkonstante ausgegeben, so können auch andere Grundtaktperiodenzeiten erreicht werden, sofern die sich ändernde Blinkfrequenz im Displaybild nicht stört (nach Reset: 40 = 0,8 Sek.). Wichtig ist noch der Hinweis, daß der SOUND-Befehl diesen CTC-Kanal zur Tondauerbestimmung nutzt und somit bei der Anwendung des vorgestellten Verfahrens in Kombination mit dem SOUND-Befehl auf die Uminitialisierung des CTC-Kanals 2 geachtet werden muß.

Dr. Dieter Döring

Literatur

/1/ Kleincomputer KC 83/3, System-Handbuch. VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen

Ermitteln des freien Diskettenspeicherplatzes mit REDABAS

Das Datenbankbetriebssystem REDABAS ist trotz mancher Nachteile gut zur Lösung von Aufgaben mit kleineren bis mittleren Datenmengen geeignet. Einer dieser Nachteile ist, daß bei erschöpfter Diskettenspeicherkapazität ohne vorherige Warnung ein Abbruch der Programmabarbeitung erfolgt. Mit dem vorgestellten Hilfsprogramm ist es problemlos möglich, die freie Speicherkapazität eines Laufwerks zu ermitteln und im Programm auszuwerten.
Bedingung dafür ist ein SCP-1715-kompatibles Betriebssystem, wobei der Laufwerkstyp beliebig sein kann, da das eingebundene Assemblerprogramm die Diskettenparametertabellen im BIOS auswertet.
Da vom Betriebssystem SCP 1715 nur 4 Laufwerke verwaltet werden, begrenzt der REDABAS-Programmteil die Laufwerksauswahl auf die Laufwerke A bis D.
Das eingebundene Assemblerprogramm kann jedoch 9 Laufwerke (A bis I) aufrufen, so daß eine Erweiterung möglich ist. Die Begrenzung auf 9 statt der zugelassenen 16 Laufwerke erfolgte, um eine möglichst einfache Parameterübergabe zu realisieren.
Bei Bedarf kann es auch in einen anderen Speicherbereich geladen werden, da nur der Aufruf der BDOS-Funktionen mit einer Absolutadresse (Call 5) erfolgt.
Mit diesen Hinweisen dürfte es einfach möglich sein, das angegebene Demonstrationsprogramm den jeweiligen Erfordernissen anzupassen.

Bernd Liebermann

```
* 12.08.87
* STAT.PRG

* Ermitteln des freien Diskettenspeicherplatzes

* Eingabe - Varialble LW (String)
*           enthaelt den Code (A..D) fuer das gewuenschte
*           Laufwerk, sonst wird das aktuelle Laufwerk
*           abgefragt
*
* Ausgabe - Variable FREE (Numerisch)
*           Sie enthaelt den freien Speicherplatz in kByte

**** Assemblerprogramm laden
SET CALL TO 42752
POKE 42752,229, 14, 25,205,  5,  0,225,245,229, 35,237,111
POKE 42764,230, 15, 95, 29, 14, 14,244,  5,  0, 14, 27,205
POKE 42776,  5,  0,229, 14, 31,205,  5,  0,229,221,225,221
POKE 42788, 94,  5,221, 86,  6, 33,  0,  0,227, 78, 35,  6
POKE 42800,  8,121,  7, 79, 56, 12,227,125,198,  1, 39,111
POKE 42812,124,206,  0, 39,103,227,122,179, 40,  6, 27,  5
POKE 42824, 40,227, 24,229,209,221, 70,  3,  4,203, 56,203
POKE 42836, 56,203, 56, 33,  0,  0,125,131, 39,111,124,138
POKE 42848, 39,103,  5, 32,245,235,221,225,221, 54,  0, 67
POKE 42860,221,114,  1,221,115,  2,241,221,229, 95, 14, 14
POKE 42872,205,  5,  0,225,201

**** Laufwerksnummer ermitteln
DO CASE
CASE 0=TEST(lw)
 STORE 0 TO free
CASE lw>='A' .AND. lw<='D'
 STORE ORD(lw)-64 TO free
CASE lw>='a' .AND. lw<='d'
 STORE ORD(lw)-96 TO free
OTHERWISE
 STORE 0 TO free
ENDCASE

**** Aufruf des Assemblerprogrammes
CALL free
STORE VAL(STR(free,4)) TO free

RETURN
```

PolyCAD

Zur Leipziger Herbstmesse 1988 wurde unter dem Namen PolyCAD ein neu entwickeltes CAD-Softwarepaket für 16-Bit-Mikrorechner vorgestellt, das im CAD/CAM-Softwareentwicklungszentrum des Kombinats Polygraph Leipzig unter Mitwirkung der TH Leipzig und des Kombinats Robotron entwickelt wird.
PolyCAD ist modular aufgebaut und besteht aus folgenden funktionellen Komponenten:
- einem relationalen Datenbanksystem
- einer leistungsfähigen Grafikschicht zur Zeichnungserstellung und zur Arbeit mit Symbolgrafiken für unterschiedliche Aufgabenstellungen
- einem Projektmanager (PM)
- einer Anzahl spezifischer Anwendermoduln, z. B. Konvertierungsmoduln, Stücklistenmodul, Informations- und Auskunftsmodul, Berechnungs- und Auswerteprogramme.

Diese funktionellen Komponenten werden in Anwenderpaketen kombiniert, z. B.:
POLYCON – Konstruktionsarbeitsplatz
Schwerpunkt: Zeichnungserstellung mit Zeichnungs- und Symboleditor sowie integrierten Datenbankfunktionen
POLYDAT – Projektierungsarbeitsplatz
Schwerpunkt: Datenbankgestützte Projektbearbeitung mit integrierten Grafikfunktionen
MBS 1700 – Modulbibliothek für Datenbankarbeit, grafische Realisierungen, nutzerspezifische Gestaltung eines Projektmanagers, Stücklistenmodul u. a.
Im Herbst 1988 ist für wichtige Systemteile die Industrieerprobung vorgesehen, der Vertrieb der Applikationssoftware wird im wesentlichen durch Robotron ab 1/89 übernommen, die Bereitstellung der Entwicklungssoftware (Modulbibliothek) erfolgt direkt durch das CAD/CAM-Zentrum Polygraph Leipzig. Informationsmaterial über PolyCAD ist ab 10/88 über das CAD/CAM-Softwareentwicklungszentrum Polygraph Leipzig, Wachsmuth Str. 4, 7031 Leipzig, Tel: 47 46 71, erhältlich.

Dr. D. Herden

CeBIT '88

Die Hannover-Messe CeBIT versteht sich als ein Weltzentrum der Kommunikations- und Computertechnik. Viele Fakten zeigen, daß die Messe diesem Anspruch in weitem Maße gerecht wird. Dies wird vor allem durch den hohen Anteil ausländischer Aussteller (USA: 174, Taiwan: 125 (1987: 67!), GB: 107) und die vielfältigen Präsentationen der Anwendungsmöglichkeiten deutlich. Überhaupt ist neben den technischen Neuerungen bei einzelnen Erzeugnissen dieser Anwendungsaspekt besonders deutlich ausgeprägt.
Auf der CeBIT '88 (16.–23. März 1988) wurden sowohl einzelne Hard- oder Software-Erzeugnisse als auch umfangreiche Systemlösungen unter anderem aus den Bereichen Büro- und Informationssysteme, Software und Unternehmensberatung, CIM, CAD, CAM und Telekommunikation vorgestellt.
Nachfolgend einige Ausführungen zu Entwicklungstendenzen und bemerkenswerten Exponaten.

Computertechnik

Der zur CeBIT '86 angekündigte Supercomputer ETA 10 der Firma Control Data Corporation wurde vorgestellt. Eine Anlage war auf dem Messestand ausgestellt. Mit diesem System wird eine Ära des Supercomputing eingeleitet. Die Systeme sollen in großen Stückzahlen produziert werden, sind vergleichsweise preiswert (Modelle P, Q, E und G zwischen ca. 1 ... 10 Mio $) und sollen vornehmlich als Abteilungsrechner im wissenschaftlichen Bereich und weniger als main frame im Rechenzentrum eingesetzt werden (Beachte: Dezentralisierungstendenz auch bei den „Großen"!). Die Zusammenstellung in der Tafel zeigt die Leistungsbreite auf.
Einiges zur technischen Basis: Es werden spezielle VLSI-CMOS-Gate-Array mit 20000 Gattern eingesetzt. Geringe Wärmeabgabe (ca. 800 W je Board) ermöglichen eine hohe Pakkungsdichte und Luftkühlung bei den kleineren Modellen. Die Stickstoffkühlung wird vor allen Dingen wegen des Geschwindigkeitsgewinns eingesetzt. Je Schaltkreis werden ca. 5000 Logikelemente für ein Selbsttestsystem eingesetzt. Alle 240 VHSI-Schaltkreise eines Zentralprozessors sind auf einer Platine mit 44 Layers untergebracht. Über 90000 durchkontaktierte Bohrungen wird das Verbindungssystem realisiert. Folglich sind im Computerschrank fast keine Kabel zu sehen. Für einen Prozessor werden nur noch 1600 W benötigt.
Eine hohe Übertragungsgeschwindigkeit zwischen Prozessor und Hauptspeicher (75 GBit/s) sichert eine kontinuierliche Datenversorgung der CPU. Alle Systemkomponenten, wie Betriebssystem, Netzwerk, Compiler und Anwendungssoftware, orientieren sich an der Hauptforderung nach einer einfachen Nutzung des Supercomputers. Es existieren Softwarekomponenten zur Kommunikation mit allen größeren Betriebssystemen von CDC oder anderer Produzenten. Für die ETA 10 selbst wird ein UNIX-kompatibles Betriebssystem angeboten. Konzepte zur Vernetzung der Supercomputer wurden vorgestellt.
Andere, ebenfalls leistungsfähige Computer wurden von Tandem, Tolerant Systems, NCR u. a. ausgestellt. Sie erreichen jedoch alle nicht diesen Durchsatz und sind vornehmlich an Einsatzfällen orientiert, bei denen es um eine hohe Systemzuverlässigkeit geht. Faktisch ausnahmslos handelt es sich dabei um Multiprozessorkonzepte.
Die nun schon „klassisch" zu nennenden EDVA waren faktisch nicht ausgestellt. Sie existieren in „irgendwelchen" Rechenzentren, ihre Ressourcen werden über Kommunikationsnetze verfügbar gemacht. Lediglich für bestimmte Einsatzgebiete, vornehmlich mit Anforderungen an erhöhte Systemverfügbarkeit, werden bestimmte Systeme konzipiert und realisiert. Andererseits gibt es Implementierungen neuer Konzepte, vor allem verschiedene RISC-Architekturen.
Es ist aber anzumerken, daß diese Klasse der EDVA zunehmend mehr bedrängt wird, von „oben" durch die Entwicklungstendenzen bei Supercomputern und von „unten" durch ständig steigende Leistungsmerkmale bei Workstations und Personalcomputern.
Bei den Minis dominieren die VAX-Systeme der Firma DEC bzw. vergleichbare Computer. Damit wird ein breites Spektrum von Leistungsmerkmalen überdeckt. Durch „Verclusterung" können außerordentlich leistungsstarke Komplexe konfiguriert werden. Die MicroVAX-Systeme werden in unterschiedlichen Formen für das Büro, die Werkhalle oder das Labor gefertigt.
Bei Workstations und Personalcomputern existiert ein außerordentlich breites Angebot. Hunderte von Firmen offerieren PCs entsprechend dem IBM PC. In den letzten Jahren hat sich jeweils ein leistungsstärkerer Typ der INTEL-8086-Familie als Schwerpunkt gezeigt. In diesem Jahr ist es die 80386–CPU, die in vielen Modellen eingesetzt ist. Ebenfalls weit verbreitet ist die 80286-CPU.

Modelle des Supercomputersystems ETA 10

ETA – 10 Modell	*P*	*Q*	*E*	*G*
Taktzeiten (ns)	*24*	*19*	*10,5*	*7*
Prozessoren	*1–2*	*1–2*	*1–4*	*2–8*
lok. HS (MByte)	*32–64*	*32–64*	*32–128*	*32–256*
globaler HS (MByte)	*64–512*	*64–512*	*256–1024*	*512–2048*
Leistung (MFLOPS)	*750*	*946*	*3.420*	*10.280*
Kühlung	*Luft*	*Luft*	*flüssiger Stickstoff*	

Seltener ist schon der einfache 16-Bit-Prozessor, faktisch ist nicht mehr anzutreffen ist eine 8-Bit-CPU in den PCs. Bei den Prozessoren dominiert deutlich die INTEL-Familie, gefolgt von den 16/32-Bit-Prozessoren von Motorola. Andere sind fast nicht anzutreffen. Des weiteren sind ein hochauflösender farbiger Bildschirm, ein Hauptspeicher mit ein und mehr MByte Kapazität und ein Massenspeicher (Harddisc) mit einigen 10 MByte Ausrüstungsstandard. Ein breites Sortiment von Ergänzungshardware (Speichererweiterungen, Kommunikationscontroller, Grafikcontroller etc.) ermöglichen nahezu unbegrenzte Einsatzfälle.
Die im Bericht über die CeBIT '87 bereits angedeutete Tendenz bei den peripheren Geräten hat sich weiter ausgeprägt. Deshalb nur einige kurze Bemerkungen.
Obwohl natürlich noch Typenraddrucker auf dem Markt sind, dominieren bei den Zeilendruckern die Nadeldrucker. Ausführungen mit 9 und vor allem 24 Nadeln sind weit verbreitet. Daß damit noch nicht das Ende erreicht wurde, demonstrierte EPSON mit einem Matrix-Drucker mit 48 Nadeln und max. 300 Z/s (siehe MP 6/88). Die damit erreichbare Schönschrift sowie die realisierte Druckgeschwindigkeit verdrängen die Typenraddrucker immer weiter. Tintenstrahldrucker mit bis zu 24 Düsen im Druckkopf, auch in Ausführungen für mehrfarbigen Druck, werden ebenfalls angeboten. Darüber hinaus etabliert sich der Laserdrucker immer mehr in diesem Bereich. Systeme mit Leistungen bis zu über 100 Seiten/min für den Massendruck, Laserdrukker mit ca. 25 Seiten/min für Anwendungen mit mittleren Anforderungen oder kleinere Modelle als Auftisch-Geräte mit 6 bis 10 Seiten/min für den Arbeitsplatzcomputer sind im Angebot. Weitere Leistungsmerkmale sind Druckgeschwindigkeit, Geräuschpegel und natürlich der Preis. Dies alles berechtigt die Aussage, daß Mosaik- und Laserdrucker in diesem Bereich dominieren werden.
Bei den Massenspeichern dominieren natürlich die mit wahlfreiem Zugriff. Nach wie vor weit verbreitet sind Floppy-Disk-Laufwerke im 5,25"-Format (slime-line-Ausführung) mit Kapazitäten bis 1,6 MByte. Erste Ausführungen von Laufwerken mit Senkrecht-Aufzeichnung bieten 4 MByte Kapazität. Des weiteren existieren natürlich Laufwerke im 3,5"-Format von mehreren Herstellern. Auch hier das Fassungsvermögen bis ca. 2 MByte. Weit verbreitet, nicht zuletzt wegen des günstigen Preis/Leistungsverhältnisses, sind Winchester-Laufwerke mit einigen 10 bis ca. 400 MByte Kapazität im 3,5"-, 5,25"-, 8"- und 9"-Format (zum Teil direkt auf der Leiterkarte bei den kleineren Formaten). Darüber hinaus gibt es Laufwerke mit größeren Kapazitäten, beispielsweise bis über 1 GByte bei 9"-Laufwerken. Angekündigt wurden 3,5"-Winchester-Laufwerke mit bis zu 200 MByte. Magnetband-Streamer mit bis zu 100 MByte Fassungsvermögen bieten Möglichkeiten zur Datensicherung bzw. -archivierung.
Es sind auch Massenspeicher mit Kapazitäten von einigen 10 bis über 100 GByte für große Informationssysteme auf optischer Basis (CD-ROM bzw. WORM (write once, read many)) im Angebot. Weitgehende Einsatzmöglichkeiten werden diese Speicher sicherlich erst finden, wenn das Problem des Mehrfachbeschreibens gelöst ist; eine Aufgabenstellung, an der intensiv gearbeitet wird.
Nachfolgend einige Bemerkungen zu Displays. Hochauflösende Bildschirme sind für Einsatzfälle des CAD oder auch des Desktop Publishing unverzichtbar. Damit werden unmittelbar Einsatzgruppen und Akzeptanzfragen derartiger Systeme entschieden. Das gilt sowohl für farbige Displays als auch für Ausführungen für Schwarz-auf-Weiß-Darstellungen. So bietet der hochauflösende Farbmonitor 7351 von Conrac Elektron mit einer 19"-Bildröhre eine Auflösung von 1280×1024 Pixel. Eine Zeilenfrequenz von 65 kHz bei max. 80 Vollbildern je Sekunde sichert ein flimmerfreies Bild. Dieses Modell hat eine Bandbreite von 110 MHz. Damit wird höchsten Anforderungen aus den Bereichen des CAD/CAM Rechnung getragen. Neben der „klassischen" Ausführungsform als CRT-Display gibt es viele Displays mit LCD-, LED- oder Plasma-Schirmen. Zweckmäßig für Bildschirm-Darstellungen in größeren Räumen sind flache Aufsätze eines Plasma-Displays für einen Overhead-Projektor, so daß PC-Ausgaben direkt auf die Leinwand projiziert werden können.

Software

Im Bereich der Workstations bzw. PCs dominieren nach wie vor UNIX (einschließlich Derivate) und das MS-DOS. Hierfür gibt es eine unübersehbare Menge von Anwender-Software. Eine gewisse Verbreitung hat das Windows 386 für die leistungsstarken PCs mit der 80386-CPU gefunden. Das OS/2 für die neue PC-Generation war erwähnt, jedoch gibt es dafür gegenwärtig noch nahezu keine Anwender-Software. Es ist damit zu rechnen, daß in kurzer Zeit die Lage völlig anders ist.
Auf die unübersehbare Vielfalt der Anwendersoftware kann hier nicht eingegangen werden. Es ist davon auszugehen, daß für alle denkbaren Einsatzgebiete im Bereich der großen Unternehmen, der mittelständischen Industrie, der Handwerksbetriebe und der Privatsphäre Software-Systeme zur Rationalisierung der informationellen Prozesse existieren.
Anwendersoftware war ein ausgesprochener Schwerpunkt der CeBIT '88. Über 100 neue Anbieter zu diesem Komplex sind Beweis dieser Einschätzung.
Neben diesen Anwendersystemen existieren vielfältige Software-Werkzeuge zur rationellen Entwicklung und Fertigung von Softwaresystemen. Diese technologischen Mittel zur Softwareproduktion gewinnen bei der sich immer deutlicher ausprägenden Anwendung der Computer einen ständig steigenden Wert.

Kommunikationstechnik

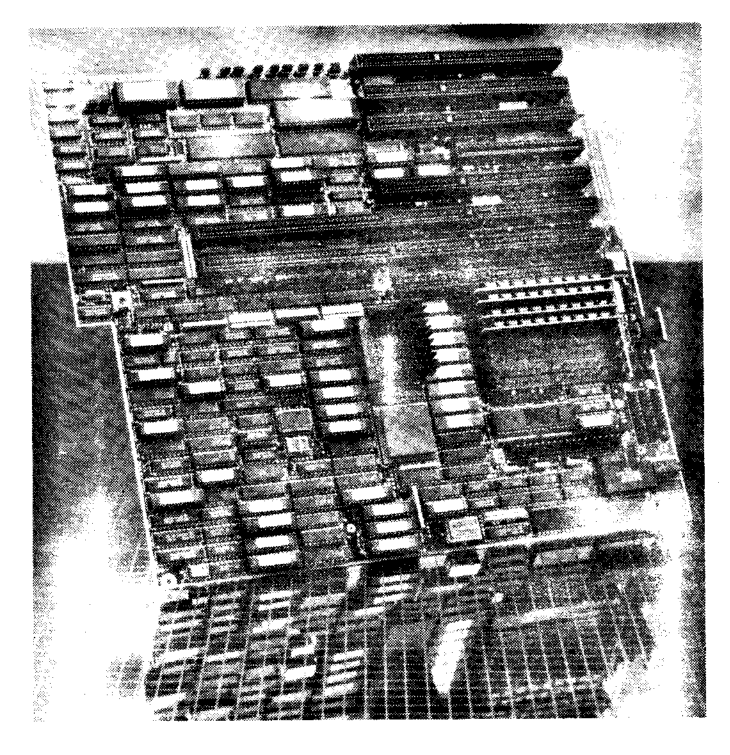
Der zweite Schwerpunkt der Messe, die Kommunikationstechnik, soll hier nur kurz vorgestellt werden.
Der Versuchsbetrieb erster ISDN-Teilnetze in der BRD und das weitere Vorankommen der internationalen Standardisierungsbestrebungen spiegelten sich bei der CeBIT '88 durch vielfältige ISDN-Präsentationen wider. Das Konzept der Bundespost sieht 1988 erste ISDN-Telefoniedienste in einigen Ballungsräumen vor. Dann sollen noch in diesem Jahr Teletex/Telex und später Telefax, Btx, Datex und Funkruf dazukommen. Dies sind für die nachrichtentechnische Industrie ohne Zweifel günstige Bedingungen, obwohl sicherlich die großen Zahlen für Anschlußeinheiten, die dann auch die Wirtschaftlichkeit (insbesondere der Rückfluß für die außerordentlich hohen F/E-Aufwendungen) gewährleisten, noch auf sich warten lassen.
Bei den Vermittlungssystemen dominieren vor allem die Angebote von Siemens und SEL (Alcatel) zum Aufbau von Orts-, Fern- und Nebenstellenzentralen. Auch in diesem Jahr war wieder deutlich der Trend zu registrieren, daß, vor allem im Nebenstellenbereich, Firmen der Computerbranche Vermittlungstechnik anbieten (IBM, Nixdorf).
Die für den Aufbau der Übertragungstrassen notwendigen Leitungstrakte, auch für die hochkanaligen Systeme, sind auf der Basis Koaxial- und vor allem Lichtwellenleiter und über Dezimeterstrecken vorhanden. Dazu kommen noch die Breitband-Kommunikationsanlagen in der neuen 450-MHz-Technik. Damit kann die bisherige Übertragungskapazität beträchtlich gesteigert werden, so daß nunmehr 35 TV-, 30 analoge und 16 digitale Hörfunk-Kanäle zum Teilnehmer geführt werden können. Dazu gehören natürlich noch technische Systeme zum Betrieb und zur Überwachung dieser Übertragungseinrichtungen.
Für den praktischen Einsatz der Lichtwellenleitertechnik ist eine zuverlässige und montagefreundliche Verbindungstechnik geschaffen worden. Dämpfungsarme Spleiß-Verbindungen, sichere Steckverbinder (Prinzip der Linsenkopplung), Muffen und Verzweiger sowie eine zugehörige Meßtechnik sind nunmehr von mehreren Anbietern verfügbar.
Die ISDN-Grundphilosophie der Diensteintegration findet natürlich auch in den Endgeräten ihren Niederschlag. Das multifunktionale Endgerät mit Bildschirm, Tastatur und Handapparat ist dabei die Basis. Es wurde ein breites Spektrum von Fernsprechapparaten, von Faksimilegeräten und Einrichtungen für den Mobilfunk gezeigt. Bei einigen Ausstellern waren Bildtelefonsysteme für den 64-KBit/s-Kanal des ISDN zu sehen. Diese extreme Datenreduktion, die dabei notwendig ist, ist vor allen Dingen bei bewegten Objekten zu sehen. Ob das eine echte Alternative zu den Breitband-Diensten des B-ISDN sein wird, scheint zweifelhaft.

Wie stark das Bedürfnis ist, PC-Technik mit ISDN zu verbinden, zeigt eine ISDN-Adapterkarte für IBM-kompatible PCs. Sie verfügt über eine S0- und eine UP0-Schnittstelle und gewährleistet die simultane Übertragung von 2 B- (2×64 KBit/s) und einem D-Kanal (16 KBit/s). ISDNCard unterstützt neben einem Komfort-Telefon ISDN-Btx, ISDN-teletex, ISDN-Telefax und ISDN-Textfax. Weitere Unterstützungen (z. B. Voicebox) sind vorgesehen.
Im Bereich der lokalen Kommunikation ergänzen sich (ISDN-fähige) Nebenstellenanlagen, die neben Telefonie vielfältige weitere Dienste der Text- und Datenkommunikation anbieten, und LAN in interessanter Weise. Natürlich sind beide Konzepte in ihren Leistungsmerkmalen nicht deckungsgleich, jedoch im Bereich der Text- und Datenkommunikation verschwinden die Unterschiede. Das Angebotsspektrum unterschied sich nicht wesentlich vom Vorjahr.

Abschließend einige Trendeinschätzungen und Schlußfolgerungen, die vornehmlich für den Bereich des Hochschulwesens gelten:
– Die Entwicklung im Bereich der Supercomputer ermöglicht vor allem in der Grundlagenforschung das Angehen völlig neuartiger Aufgabenklassen. Deren Lösung verspricht nicht nur Erkenntnisgewinn schlechthin, sondern schafft wissenschaftlichen Vorlauf für eine leistungsstarke Volkswirtschaft der nächsten Jahrzehnte. Die Aussage, die auch in den Ausrüstungskonzeptionen des Ministeriums für Hoch- und Fachschulwesen der DDR zur Rechentechnik an den Universitäten und Hochschulen ihren Niederschlag gefunden hat, kann nur immer wieder hervorgehoben werden.
– Die Dezentralisierung der Computertechnik schreitet auch bei den „Großen" immer weiter voran. Die Leistungsmerkmale im Bereich der PC und der Workstations einerseits und die Preisentwicklung, auch im Bereich der Supercomputer, andererseits lassen daran keinen Zweifel mehr. Dies sollte Anlaß sein, die Aufgaben und Stellung der Rechenzentren an den Universitäten und Hochschulen neu zu durchdenken und dieser Entwicklung anzupassen.
– Die Anwendung der Computertechnik in allen Bereichen erfordert einen angemessenen Bildungsvorlauf. Die Maßnahmen zur Ausgestaltung der Informatik-Ausbildung im Hochschulwesen entsprechen dieser Entwicklung. Stärker als bisher sind die Anwendungsaspekte und das Erfordernis der Analyse der durch Computereinsatz zu automatisierenden Prozesse zu betonen.
– Das Angebot an Publikationen zur Informatik und der Anwendung der Computer ist in der DDR nach wie vor unzureichend. Das betrifft sowohl den Buch- als auch den Zeitschriftenmarkt.
– Die Forschungsaufgaben zur Kommunikations- und Computertechnik an den Universitäten und Hochschulen erfordern einen ständig steigenden materiell-technischen Aufwand. Besonders deutlich und hemmend zugleich zeigt sich das bei Leiterkartenentwurf und -fertigung. Praktische Aufbauten mit VLSI-Schaltkreisen erfordern zwingend Mehrebenenleiterkarten mit durchkontaktierten Verbindungen und eine angemessene Meß-, Registrier- und Darstellungstechnik (z. B. Logikanalysatoren). Die notwendigen Investitionen können ohne Zweifel nur noch im Zusammenhang mit einer sinnvollen Konzentration erbracht werden.

Prof. Dr. sc. techn. Neubert

Hannover-Messe Industrie 1988



Auf der Hannover-Messe Industrie 1988 (19.–26. April 1988) waren 5800 Aussteller aus 43 Ländern vertreten. Insgesamt wurden etwa eine halbe Million Besucher gezählt.
Die Palette der gezeigten Exponate reichte von Mikroelektronik-Bauelementen über Rechnersysteme und Automatisierungsanlagen bis hin zur Weltraumtechnik.
Breiten Raum nahm die Vorstellung neuer Technologien und der damit zusammenhängende Lizenzhandel (Know-how-Transfer) ein. Beherrschendes Thema eigentlich auf fast allen Gebieten war der Siegeszug der Mikroelektronik in Dimensionen, wie sie dem sonst wohl immer auf einem speziellen Fachgebiet tätigen Besucher bei solchen Gelegenheiten besonders deutlich werden.
Für das Fachgebiet der Mikroelektronik und Mikroprozessortechnik sind die folgenden, jeweils zu Themengruppen zusammengefaßten Sonderschauen besonders erwähnenswert:

- MICROTRONIC – Bausteine der Elektronik
- INTERMATIC – Fertigungsnahe Computertechniken
- ICA – Internationales Zentrum für Anlagenbau
- Innovationsmarkt Forschung und Technologie

Die DDR war in Hannover 1988 in vielfältiger Weise präsent. Zum neunundzwanzigsten Mal zeigte eine Reihe von Kombinaten und Außenhandelsbetrieben aus der DDR ihr gestiegenes Leistungsvermögen. Die Bedeutung, die dem Auftreten der DDR auf diesem Welthandelsplatz beizumessen ist, wurde nachdrücklich durch den Besuch wichtiger Regierungsdelegationen aus der DDR unterstrichen.
Auf zwei besonders wichtige Schwerpunkte des Ausstellungsgeschehens in Hannover – ASICs und MAP – soll hier näher eingegangen werden.
Auf der MICROTRONIC-Sonderschau versuchten die Hersteller von

Bild 1 Von der Firma Intel wurde erstmals das System 302, ein auf dem AT-Bus basierender 25-MHz-80386-Personal-Computer, vorgestellt. Er ist weltweit mit 25 MHz einer der schnellsten 80386-PC und setzt damit in verschiedener Hinsicht neue Maßstäbe (das Bild zeigt das Motherboard).
Die Zielmärkte des Systems 302 sind zeitkritische Anwendungen wie Echtzeitprozeßkontrolle, Simulation, Computer Aided Design und der Bankenbereich – alles Anwendungen, bei denen primär hohe Verarbeitungsgeschwindigkeit und große Speicherkapazität die Hauptanforderungen sind.

Bauelementen, Baugruppen und Systemen der modernen Elektronik zu vermitteln, wie die Schlüsseltechnologie Mikroelektronik zur zentralen Bedeutung für viele Anwender wird. Es wurde dabei insbesondere versucht, zu zeigen, welche Applikationshilfen dem Anwender zur Verfügung stehen, um seine Produkte auf den neuesten Stand mikroelektronischer Leistungsfähigkeit zu bringen. Wichtige Schwerpunkte waren die drei Themenkomplexe unter dem Motto

- Vom Bauelement zur Systemintegration
- Technologiefortschritt schafft neue Strukturen
- Elextronic Design Automation – Schlüssel zur Systemintegration

mit praktischen Beispielen aus allen Bereichen der Elektroindustrie, von der Meß-, Steuer-, Regeltechnik bis hin zur Preßautomation.

Eine besonders wichtigen Schwerpunkt des Angebotes bildeten die anwendungs- und kontinuierten Schaltungen, die ASICs (Applications Specific Integrated Circuits).

Ihnen stehen die Standardschaltungen gegenüber, die einheitlich gefertigt werden. Standardschaltungen sind neben den TTL/ECL/CMOS-Standardbauelementreihen vor allem die Speicher, Mikroprozessoren und Microcontroller, Operationsverstärker, Signalwandler von analog zu digital und umgekehrt usw.

Während die Standardschaltungen im kapitalistischen Wirtschaftsgebiet nahezu ausschließlich von Japan (54 %) und den USA (44 %) gefertigt werden, stammt jede vierte nichtstandardisierte (ASIC-) Schaltung aus Westeuropa (26 %). Die Marktbedeutung der ASICs nimmt ständig zu.

In den verbleibenden zwölf Jahren bis zum Ende dieses Jahrhunderts wird sich nach Schätzungen in der BRD der ASIC-Verbrauch mindestens versiebenfachen.

Das Verhältnis von Standard zu Nichtstandard auf dem IC-Gebiet wird sich bis dahin, von jetzt noch zwei zu eins, auf eins zu eins zugunsten der ASICs gewandelt haben.

Das überproportionale Wachstum der ASIC-Schaltungen ist das äußere Zeichen ihrer Leistungsfähigkeit – maßgeschneidert für eine spezielle Aufgabe im Gerät oder in der Anlage zu sein.

Die Leistungsfähigkeit der ASICs und ihre Anwendungsmöglichkeiten zu vermitteln, war ein besonderes Anliegen der Hannover-Messe.

Die Integration der Computertechnik in die hochinnovative Automatisierungstechnik der Industrie war kein punktuelles Thema einiger Spezialisten, sondern ein übergreifendes Leistungsverbot vieler Aussteller.

Konzentrationspunkt war die Teilfachmesse INTERMATIC, in der 200 Aussteller (u. a. auch aus der DDR) CIM-Konzepte praxisnah darzustellen versuchten (CIM – Computer Integrated Manufacturing).

Im Brennpunkt dieser Präsentation stand ein Leitzentrum „CIM – Fabrik mit Zukunft".

Hier führten einige Unternehmen vor, wie auf der Basis von Standards und definierten Schnittstellen zukunftsorientierte computergestützte und -integrierte Systeme schon heute mit vorhandenen Anlagen und Strukturen realisiert werden können. Ihre unterschiedlichen Computersysteme, Produktionsstationen, Maschinensteuerungen und Kommunikationsrechner waren alternativ oder in Kombination über ein Breitband-Netzwerk bzw. ein Lichtwellenleiter-Netzwerk miteinander verbunden.

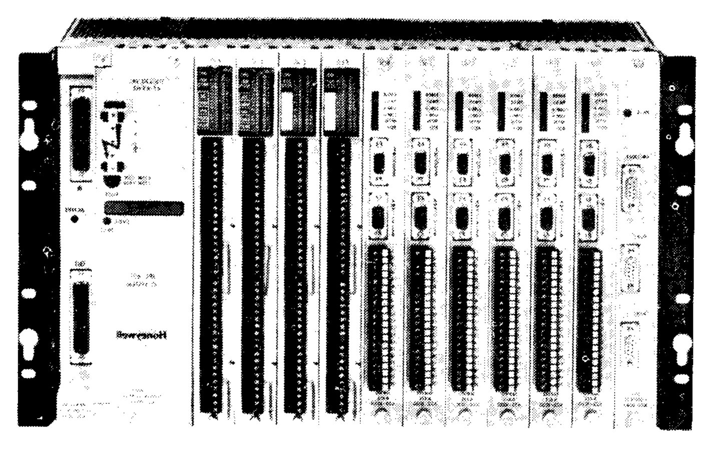


Bild 3 Von Honywell wurde das vielfältig konfigurierbare Industrieautomatisierungssystem IPC 620/622 auf der Basis des Mikroprozessors M 68008 mit einem tragbaren Programmiergerät (MS-DOS) gezeigt.

Darüber hinaus unterstrichen alle Teilnehmerfirmen ihre Bereitschaft zu offenen Systemen, genormten Schnittstellen und Standards.

Detailliert wurden mehrere praxisnahe Bereiche der Produktionsautomatisierung vorgestellt.

Besonderes Interesse fanden offene, herstellerübergreifende Kommunikationslösungen als Basis für integrierte Lösungen im Fertigungsbereich.

Das „Manufacturing Automation Protocol" (MAP) will international hier eine Normierung festlegen.

Dadurch soll eine der Grundlagen für CIM geschaffen und die notwendige Sicherheit im Automatisierungsbereich hergestellt werden.

Ein wichtiges Kennzeichen von MAP war, daß Anwender und Hersteller in gemeinsamen Anstrengungen über Firmengrenzen hinweg genormte Lösungsansätze spezifizierten. Gerade die Forderung der Anwender nach genormten Kommunikationsprodukten muß als besonders wichtig angesehen werden. MAP soll eine Antwort sein auf die schwierigen Kommunikationsbeziehungen verschiedener Automatisierungssysteme wie Werkzeugmaschinen, Roboter, Computer und speicherprogrammierbare Steuerungen. MAP spezifiziert, wie die Kommunikationsanforderungen in der Fertigungsautomatisierung ohne herstellerspezifische Einschränkungen bewältigt werden können. Dabei muß die durch MAP festgelegte Kommunikationsarchitektur allgemeinen Anforderungen genügen, das heißt, alle typisch auftretenden Anforderungen weitestmöglich abdecken und dabei innerhalb der genormten Architektur heutige und künftige Strukturen berücksichtigen, ohne technische Weiterentwicklung zu behindern.

Die heute aktuelle Version von MAP ist die Version 2.1. Auf dieser Basis gibt es Produkte einiger weniger Hersteller. In den Normungsgremien wird zur Zeit an der nächsten Version MAP 3.0 gearbeitet; sie soll für die nächsten sechs Jahre stabil bleiben.

Die Firma ALLEN-BRADLEY demonstrierte in Hannover zum ersten Mal das „MAP-3.0-Starter-Paket". Im Rahmen der universellen Kommunikation unterschiedlicher Hersteller, die über das Netz Daten austauschen, wurde dabei ein MAP-3.0-Produkt vorgestellt, das der im Juni 1988 verabschiedeten MAP-3.0-Version entsprechen soll.

Das „MAP-3.0-Starter-Paket" wurde als Subnetzwerk zur Breitband-Kommunikation installiert und lauffähig mit einer speicherprogrammierbaren Steuerung PLC-3 gezeigt. Die PLC-3-MAP-Schnittstelle paßt dabei in einen freien Steckplatz der PLC-3 und erlaubt unterschiedlichen Datenaustausch: Lesen eines speziellen Files von einer anderen Station, Empfangen von Prozessorzustandsänderungen, Austausch von kompletten Programmen und Diagnosen über angeschlossene Stationen.

Dr. Ludwig Claßen

256-KByte-RAM-Erweiterung

Das Kombinat PRÄCITRONIC Dresden beabsichtigt, im Rahmen seiner Konsumgüterproduktion eine Leiterplatte (zweiseitig) zur 256-KByte-RAM-Erweiterung (entsprechend der Veröffentlichung in der Zeitschrift Mikroprozessortechnik 3/88) zu produzieren.

Interessenten richten bitte ihre Anfragen an:
VE Kombinat PRÄCITRONIC
Dresden Stammbetrieb, Abt. KM,
Fetscherstr. 72, Dresden, 8016

In eigener Sache

Für eine weitere Redakteurstelle in unserer Zeitschrift suchen wir einen geeigneten Mitarbeiter bzw. eine Mitarbeiterin mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik.

Zu den Aufgaben gehören unter anderem das redaktionelle und fachliche Bearbeiten von Manuskripten, der Besuch und die Auswertung von Fachtagungen, -messen und -ausstellungen, die Zusammenarbeit mit Autoren und Gutachtern sowie ggf. das Testen und Beurteilen von Programmen, die der Redaktion zur Veröffentlichung eingereicht werden.

Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, schreiben Sie uns oder rufen Sie uns unter Tel. 2 87 03 71 oder 2 87 02 03 an.

Das Schulungszentrum des Büromaschinenwerkes

Sömmerda führt Softwareschulungen 1988 und 1989 durch für:

- Betriebssystem DCP 3.2 (1 Woche)
- REDABAS (2 Wochen)
- T-PASCAL (3 Wochen)
- MULTICALC (1 Woche)

Anfragen sind zu richten an:
VEB Robotron Büromaschinenwerk „Ernst Thälmann" Sömmerda, Abt. Schulungszentrum, Weißenseer Str. 52, Sömmerda, 5230; Tel.: 4 25 85

Krauße

TERMINE

33. Internationales Wissenschaftliches Kolloquium

WER? Technische Hochschule Ilmenau
WANN? 24. bis 28. Oktober 1988
WO? Technische Hochschule Ilmenau
WAS? Reihe B: technische und angewandte Informatik/Computertechnik
WIE? Teilnahmewünsche an TH Ilmenau, 33. IWK, PF 327, Ilmenau, 6300; Tel. 7 48 46

Prof. Dr. Dr. Roth



Literatur

Kleines Lexikon der Speichertechnik

Von H. Völz, VEB Verlag Technik, Berlin 1987, 68 Seiten, 48 Bilder, 4,80 M

Dieses kleine Lexikon beginnt mit einführenden Hinweisen und konzentriert sich stark auf die gebräuchlichen Termini der magnetomotorischen Speichertechnik.
Mit Sorgfalt wurden ca. 500 Fachtermini aus dem Problemkreis der Speichertechnik ausgewählt, wobei 209 Fachtermini (Ober- und Sammelbegriffe) mit ausführlichen Texten versehen sind und die restlichen als Verweistermini auftreten.
Der Text zu den Ober- bzw. Sammelbegriffen beginnt stets mit einer knappen Erläuterung (Definition) des Begriffes selbst. Daran schließen sich weiterführende Erläuterungen an, die teilweise sehr detailliert sind (insbesondere die Begriffe der magnetomotorischen Speichertechnik betreffend). Das Verständnis der angegebenen Erläuterungen wird durch eine große Anzahl von Bildern, Schaltungsstrukturen, Diagrammen und Tabellen stark gefördert. Die Begriffe sind gut ausgewählt und systematisch beschrieben. Trotzdem bleiben dabei natürlich auch Wünsche offen. Das sind insbesondere Begriffe, den Problemkreis der Halbleiterspeichertechnik betreffend, bei dem der Autor bewußt Lücken in Kauf genommen und auf entsprechende Literatur verwiesen hat.
Beispielsweise erscheint bei dem Begriff *Cache* der Verweis auf den Begriff *Seitenadressierung* als eine etwas zu einseitige Betrachtung. Das gleiche gilt auch z. B. für den Begriff *Datenblock,* der nur im Zusammenhang mit dem Magnetband erläutert wurde. Die Aussage, daß Halbleiterspeicher in Rechnern meist als Arbeitsspeicher genutzt werden, ist eventuell auch etwas zu eng gefaßt. Wünschenswert wäre ebenfalls, wenn bei allen Verweisstichworten, die eine Abkürzung darstellen (z. B. BBD, CAS, CE, ..., DMS, ds, ..., LIFD, ..., ZDMS, ZMD), die entsprechende Bezeichnung konsequent ausgeschrieben wäre. Entsprechende Beispiele lassen sich im Lexikon ebenfalls zahlreich finden (z. B. DAT, DM, ..., ECC usw.).
Die Handhabung des Lexikons wird durch die übersichtliche Darstellung und den Fettdruck der Stichworte gut unterstützt. Es ist für viele Leser von Interesse, sowohl für Studierende an Hoch- und Fachschulen als auch für Praktiker. Dieses Lexikon bietet die Möglichkeit, Grundwissen bezogen auf die Speichertechnik wesentlich zu erweitern (magnetomotorische Speicher), aufzufrischen bzw. zu ergänzen. Es kann dem Leser als gutes und fachkundiges Nachschlagewerk sehr empfohlen werden.
Eine Erweiterung auf 80 Seiten in Verbindung mit einer Neuauflage ist zu wünschen (Speicheradressierungen, Speicherverwaltung, objektorientierte Speicherung).

Dr. Veronika Sauer

Fertigung, Prüfung und Montage von Automatisierungsanlagen mit Mikrorechnern

von G. Venus und H. Beucher
Schriftenreihe „Automatisierungstechnik", Heft 18, Betriebssektion der KDT und Bereich Technologie des VEB Geräte- und Regler-Werke „Wilhelm Pieck" Teltow, 1987, 60 S., 13 Abb., broschiert 8,– M.

Anhand des mikrorechnergestützten Prozeßleitsystems audatec, das vom VEB GRW „Wilhelm Pieck" Teltow für die Automatisierung der verschiedensten verfahrenstechnischen Anwendungsfälle geschaffen wurde, gehen die Autoren der Broschüre auf die Schwerpunkte der Fertigung, Montage und Inbetriebnahme von bestimmten audatec-Komponenten ein.
Im Rahmen der technologischen Konzeption der stationären Fertigung werden eingehend der technologische Fertigungsumfang und -ablauf, die Fertigungsorganisation, die Gefäßfertigung und insbesondere die Prüfung von Baugruppen, Gefäßen und Anlagenteilen beschrieben. Die Montage von Automatisierungsanlagen, die den Fertigungsumfang und -ablauf sowie die Arbeitsvorschriften der Anlagenmontage umfassen, bilden den zweiten Schwerpunkt.
Maßnahmen der Inbetriebnahme von audatec-Automatisierungsanlagen sowie die Anforderungen an die Transport- und Lagertechnik werden kurz angedeutet. Es werden ausschließlich technische Gesichtspunkte des Produktionsprozesses von Automatisierungsanlagen dargestellt, wobei die Vernetzung mit vorbereitenden Prozessen wie Materialwirtschaft bzw. mit betrieblichen Planungs- und Abrechnungsprozessen vernachlässigt wird.
Anhand der Abbildungen werden die einzelnen Prozeßabschnitte der Fertigung, Prüfung und Montage von Automatisierungsanlagen mit Mikrorechnern anschaulich dargestellt. Diese Veröffentlichung ist insbesondere für Technologen, Projektanten, Prüfingenieure und Inbetriebnahmepersonal in der Automatisierungstechnik als Arbeitsmittel besonders gut geeignet.
Die Broschüre ist über die Betriebssektion der Kammer der Technik oder über die Zentrale Informationsstelle Wissenschaft und Technik des VEB GRW „Wilhelm Pieck" Teltow, Oderstr. 74/76, Teltow, 1530, zu beziehen.

Blackert

Schaltnetzteile

von J. Jacob, Militärverlag der DDR, Berlin 1987, 128 Seiten, 151 Bilder, 32 Tabellen, 6,– M

Moderne Stromversorgungseinrichtungen basieren auf dem Prinzip des Schaltspannungsreglers. Neben einem besseren Wirkungsgrad gegenüber konventionellen Netzteilen zeichnet sich das Schaltnetzteil auch noch durch geringeren Materialeinsatz aus. Dieser Vorteil wird zwar durch eine komplizierte Schaltungstechnik erkauft, da aber die Ansteuerschaltungen des Schaltreglers als Schaltkreis realisiert werden, spielt der Schaltungsaufwand heute keine Rolle mehr. So werden Schaltnetzteile in der Elektronik zunehmend eingesetzt. Zum anderen wendet sich auch der Amateur immer häufiger dem Entwurf und Aufbau einer derartigen Stromversorgungstechnik zu.
Es ist also notwendig, die Wirkprinzipien des Schaltnetzteils zu kennen. Der vorliegende Band will den interessierten Leser sowohl theoretisch als auch praxisnah und experimentell an die Problematik heranführen.
Über die physikalischen Grundlagen, Auswahl- und Einsatzkriterien der wichtigsten Bauelemente, Funkentstörung und sehr ausführlich diskutierte Dimensionierungsbeispiele wird der Leser schließlich zu zwei einfachen, im Schaltungsaufbau übersichtlichen und einem anspruchsvolleren Applikationsbeispiel geführt.
Ein kurzer, aber durchaus notwendiger Abschnitt vermittelt Aussagen zum internationalen Stand und zu etwaigen Tendenzen. Da das Buch versucht, möglichst umfassend über die wichtigsten Bauelemente zu informieren, schließt sich auch ein zwar kleiner, aber dem Charakter des Buches zuträglicher Tabellenanhang an. Dem aufmerksamen Leser vermitteln schließlich die Literaturangaben, daß es weitere allerdings meist schwer zugängliche Literatur gibt.
Damit ist das vorliegende Buch für alle an dieser Technik Interessierten ein echtes Hilfsmittel.

Dr. I. Schreiber

Mikrocomputergrafik

(Mikrokomputernaâ grafika)
von D. Hearn und M. Baker, in russischer Sprache, Übersetzung aus dem Englischen, Verlag Mir Moskau 1987, 352 Seiten mit Bildern, Kunstleder 10,– M

Die grafische Darstellung spielt bei der Anwendung moderner Rechner eine entscheidende Rolle, denn sie gestaltet den Dialog Mensch-Maschine anschaulicher.
Das Buch ist in fünf Kapitel mit folgendem Inhalt gegliedert: 1. Einleitung über Computergrafik; 2. Grundlagen der Computergrafik; 3. Grafische Methoden mittleren Schwierigkeitsgrades, wie z. B. Vergrößern, Verkleinern, Drehen, bewegte Bilder und Fenstertechnik; 4. Dreidimensionale Darstellungen ; 5. Anwendung von Computergrafiken, wie Leitung und Planung, Lehrmethoden, Budgetplanung, Spiele.
Die Programme sind in BASIC angegeben, und von dem Computer wird Vollgrafik und Farbtüchtigkeit vorausgesetzt. Die Problematik wird nur vom Standpunkt der Software betrachtet. Zahlreiche Beispielprogramme und Aufgaben zur selbständigen Lösung bereichern den Inhalt. Zu den Aufgaben werden Lösungshilfen angeboten.
Das Buch ist auch farbig illustriert und umfaßt eine Vielzahl von Problemen aus allen Bereichen der Technik, Wissenschaft und der Unterhaltung. Die Programme können übernommen werden oder dienen der Anregung für Erweiterungen. Es wird daher ein breiter Leserkreis angesprochen, dem das Buch eine wertvolle Hilfe zur Lösung von Grafikaufgaben bietet.
Nicht hoch genug kann das Verdienst des Verlages eingeschätzt werden, das Buch übersetzt und herausgebracht zu haben.

Dr. H. Dobesch

Bejsik dlja PK IBM (BASIC für IBM-PC)

von H. Morrill, Übersetzung aus dem Englischen, 1. Auflage, Moskva: Finansy i statistika 1987, 207 Seiten, 5,50 M

Mit der zunehmenden Verbreitung von 16-Bit-Mikrorechnern in unserer Republik ist ein gesteigertes Interesse an Software für IBM-Computer und zu diesen kompatiblen Rechnern zu verzeichnen.
Das vom Moskauer Verlag „Finanzen und Statistik" vorgelegte Buch versteht sich in erster Linie als Anleitung zum Erlernen von BASIC. Obwohl diese 1964 entstandene Sprache heute in der Rechnerarbeit als nicht ausreichend effizient eingeschätzt wird, dürfte ihr Nutzen vor allem für Einsteiger auf dem Gebiet der Rechentechnik unumstritten sein. Diesem Gedankengang folgen offenbar auch die Computerhersteller, indem sie selbst für ihre neuesten PC-Entwicklungen BASIC-Interpreter bzw. -Compiler anbieten.
Manches BASIC-Lehrbuch wird nicht nur zum Erlernen gerade dieser Programmiersprache gekauft, sondern gleichzeitig auch als populäre Literatur für Informatik-Anfänger genutzt. Ganz in diesem Sinne vermittelt „BASIC für IBM-PC" auch Grundlagenkenntnisse über Informationsverarbeitung, den Aufbau eines Computers, über Zahlensysteme usw.
Das Buch umfaßt 14, in fünf Hauptteile gegliederte Kapitel. Neben der Vorstellung des BASIC-Sprachumfangs (der übrigens am Ende der Broschüre in tabellarischer Form nochmals mit Erläuterungen aufgelistet ist), werden auch spezielle Probleme der File-Arbeit behandelt. Die einzelnen Kapitel beinhalten jeweils Übungsaufgaben, deren Lösungen am Schluß des Buches vorgestellt werden. Die sowjetische Ausgabe enthält eine zusätzlich angefügte Liste mit Kommentaren zu Fehlerausschriften, eine chronologische Darstellung der Entwicklung der Sprache BASIC sowie eine Gegenüberstellung des Sprachumfanges der BASIC-Versionen verschiedener Rechner. Kritisch anzumerken sind die handschriftlich vorgenommenen Korrekturen einzelner der im Buch zahlreich verwendeten Rechnerausdrucke, bei denen offensichtlich stellenweise das Farbband abgenutzt war. Dies mindert jedoch nicht im geringsten den positiven Gesamteindruck dieses nützlichen, insbesondere für Hochschulstudenten, aber auch andere PC-Nutzer vorgesehenen Werkes.

G. Mau

Arithmetikprozessor unter DOS 3.20

Während der Arbeit mit TURBO-PASCAL-, C- und FORTRAN-Programmen unter DOS 3.20 kam es besonders an Stellen, an denen arithmetische Ausdrücke mit Unterstützung des Ko-Prozessors berechnet werden sollten, zu unkontrollierten Systemabstürzen. Nach intensiver Suche fand sich die Ursache in einem Betriebssystemfehler des DOS Version 3.20. Eine nicht korrekte Verwaltung des Arithmetikstackpointers führt hier schnell zu einem Overflow der 8 Stackebenen. Dieser Effekt wird auch durch die Literatur bestätigt.
Eine Änderung im DOS 3.20 behebt die Fehlerursache. Die geänderte Version wurde von uns auf verschiedenen IBM-kompatiblen Systemen mit 8087 erfolgreich getestet.

Karl-Marx-Universität Leipzig, Sektion Physik, Karl-Marx-Platz 10, Leipzig, 7010

Scheller

Hilfsprogramme

Zur Verbesserung der Programmierarbeit wurden von uns folgende nachnutzbare Hilfsprogrammme erarbeitet:
• Rollen Bildschirmbereich unter REDABAS
• Druck Bildschirminhalt (Hardcopy) unter REDABAS
• Lochbandverarbeitung (R300/ISO Code) unter REDABAS
Direkte Eingabe der Lochbanddaten in REDABAS oder Text-Datei.
Die Lochbandausgabe erfolgt über eine Stringvariable und kann beliebige Trenn- und Endezeichen enthalten.
• Bildgenerator für TOURBO-PASCAL
Es werden ganze Bildschirmbilder in Dateien gespeichert. Aufbau, Änderung, Druck und Kopie der Bilder erfolgen menügesteuert. Das Gestalten der Bildschirmbilder wird wesentlich erleichtert und erfolgt ohne Änderung im Anwenderprogramm. Für die Anzeige der Bilder in Anwenderprogrammen ist lediglich ein Procedure-Aufruf erforderlich.

VEB Mülana-W3, Abt. EDV, Kirchstraße 10–12, Ohrdruf, 5807, Tel. 22 01 App. 21

Schauer

Kopplung zwischen den Kalkulationsprogrammen der 8- und 16-Bit-Technik

Am Institut für sozialistische Wirtschaftsführung der Leichtindustrie an der Hochschule für Ökonomie „Bruno Leuschner" Berlin wurde ein Programm entwickelt, das die Übertragung von Tableaudateien aus den 8-Bit-Programmsystemen KP oder SuperCalc (*.CAL) in Tableaudateien des 16-Bit-Programmsystems TABCALC (*.TCT) und umgekehrt realisiert. Das Programm ist primär darauf ausgerichtet, Formelsysteme zwischen den Programmsystemen zu übertragen, kann aber auch zur reinen Datenübertragung genutzt werden.
Das menügesteuerte Programm läuft auf dem PC 1715 und dem A 7100 unter Steuerung des jeweiligen Betriebssystems SCP.

Hochschule für Ökonomie „Bruno Leuschner", Institut für sozialistische Wirtschaftsführung der Leichtindustrie, Forschungslabor, An der Schleuse 3–4, Woltersdorf, 1255

Prof. Dr. Richter

VIDEO-Steuergerät

Das Steuergerät ermöglicht den Anschluß handelsüblicher Fernsehgeräte als zusätzliche Monitore an Computern der Typen A 5120/A 5130, PC 1715, A 7100 für Schulungs- und Demonstrationszwecke. Dadurch können die Bildschirminformationen einer größeren Personenzahl zugänglich gemacht werden.
Es sind bis zu 7 Zusatzmonitore anschließbar. Die Stromversorgung des Steuergerätes erfolgt durch den jeweiligen Computer. Eine Variante für IBM-kompatible Technik ist in Vorbereitung.

ZBE Landbau Meißen, Bohnitzscher Straße 14, Meißen, 8250

Dr. Schiller/Körner

Betriebssystem CP/K 86 für AC A 7100

Das System CP/K 86 ist kompatibel zum System CP/M 86 und weitestgehend kompatibel zum SCP 1700.
In der vorliegenden Version 2.2 unterstützt das System zur Zeit:
– 2 Diskettenlaufwerke 80spurig, doppelseitig
– 1 Drucker mit Centronics-Interface.
Der Tastaturtreiber hat zwei Modi. Im Modus 0 werden die Escape-Folgen analog zu SCP 1700 realisiert. Im Modus 1 werden die Sondentasten umkodiert. Die Kursor-Tasten liefern dann die von 8-Bit-Rechnern gewohnten Zeichen. Die PF-Tasten sind mit Nutzerstrings belegbar.
Der Kaltstart erfogt von Disketten im Format 780 KByte. CP/K erkennt Disketten mit 148, 190, 200, 624, 780 und 800 KByte automatisch beim Login.
Zum System gehören:
– das Programm FORMAT (Initialisieren und physisches Kopieren von Disketten) mit deutscher Nutzerführung
– das Programm CPKKEY (zur Belegung der PF-Tasten mit Nutzerstrings)
– die Dokumentation CPK.DOC.

VEB Energiekombinat Berlin, Stammbetrieb, Direktionsbereich Investitionen und Anlagenbau, Littenstr. 109, Berlin, 1026

Honka

Prädikat- und Syntaxbeschreibung für TURBO-PROLOG

Zukünftig nehmen die Probleme der Wissensverarbeitung in Produktion, Entwicklung und Forschung einen immer breiteren Raum ein. An die Programmieung werden neue Anforderungen gestellt, die mit den konventionellen algorithmischen Sprachen nur tieleweise lösbar sind.
Der Personenkreis, der sich mit der Programmierung mittels KI-Sprachen wie PROLOG und LISP beschäftigt, nimmt zu und wird sich weiter vergrößern.
TURBO-PROLOG ist eine PROLOG-Version, die unter dem Betriebssystem MS-DOS läuft und für Einsteiger besonders geeignet ist. Der problemlose Wechsel von Editor, Compiler und Tracer wird durch Window-Technik unterstützt und erleichtert das Erlernen dieser neuen Programmierdenkweise.
Am CAD/CAM-Zentrum unserer Universität wurde eine Prädikat- und Syntaxbeschreibung erarbeitet, die ca. 30 Seiten umfaßt und als Diskettentextdatei vorliegt.
Die Standardprädikate sind übersichtlich systematisiert und durch ein einheitliches Beschreibungsschema dargestellt, das die Argumente, Fließmuster und Kurzbeschreibung beinhaltet. Eigene Testerfahrungen sind in die Beschreibung eingeflossen.

Technische Universität „Otto von Guericke", CAD/CAM-Zentrum, PSF 124, Magdeburg, 3010

Prof. Dr. Klaeger/Springer

BUS-SKR im Vertrieb

Seit Anfang 1988 wird die SKR-Version des Basisübungs-Systems (BUS) ausgeliefert. BUS-SKR ist aufwärtskompatibel zu den ESER-Versionen BUS-TSO sowie BUS-SVM und wird bereitgestellt für die Betriebssysteme MOOS 1600, OS/RW und MUTOS 1630
Die Nachnutzungskosten betragen je nach Umfang ca. 3 TM. Damit erweitern sich die Anwendungsmöglichkeiten dieser Basis-Teachware auf die Rechner K 1630, CM-4, CM-14, I-100 und CM-52.
Versionen für BC/PC, AC und ESER-PC befinden sich in Vorbereitung.

VEB Weiterbildungsakademie, Schulungszentrum, Am Rechenzentrum, Fürstenwalde, 1240

Schubert

Testmonitor für Kleincomputer

Von uns wurde ein residenter Testmonitor für die Bearbeitung von Maschinenprogrammen auf Z 9001, KC 85/1 und KC 87 entwickelt.
Er dient zum Testen und zur Fehlersuche oder Analyse von vorhandenen Maschinenprogrammen, ist aber auch für die Entwicklung von Maschinenprogrammen oder für Untersuchungen des Mikroprozessorsystems U880 geeignet.
Merkmale:
– Screeneditor zur rationellen Befehlseingabe
– virtueller Registersatz
– Step-Routine
– Abarbeitungsroutinen mit 1 Breakpoint
– Line-by-line-Assembler (ohne Marken)
– Disassembler
– Copy-Routine für simultanes Kopieren von Kassette zu Kassette (auch Programme, die mehrteilig, kopiergeschützt oder logisch nicht einladbar sind)
– komfortable Load- und Save Routine
– Routinen zur Speichermanipulation (Anzeigen, Verändern, Füllen und Verschieben von Speicherbereichen, Suchen nach Bytefolgen)
– Peripherie-Ein-und-Ausgabe-Routine.
Der Monitor existiert in 3 Versionen, jeweils für das Ende der RAM-Module (2400-3FFFH, 6400-7FFFH, A400-BFFFH).

Computerclub der Spezialschule „Heinrich Hertz", math.-naturwiss.-tech. Richtung, Frankfurter Allee 14 a, Berlin, 1035

Automatische Erstellung Inhaltsverzeichnis

Bei der Abfassung umfangreicher Dokumentationen mittels Textprozessor TP auf PC 1715 u. a. stellt sich die Anfertigung des Inhaltsverzeichnisses als notwendiger, jedoch lästiger Routineprozeß dar. Insbesondere dann, wenn aus redaktionellen Gründen eine Neuformatierung der Textabschnitte erfolgt bzw. das Einfügen von Textstellen die ehemalige Seitennumerierung verschiebt.
Besser ist es, sich jetzt der Zusatzkomponente IVZ, in TURBO-PASCAL geschrieben, zu bedienen.
Vorerst läßt man das Inhaltsverzeichnis bei der Texterfassung außer acht. Im nachhinein analysiert das Programm IVZ entsprechend dem Reglement zur Ausdeutung von Überschriften den Text und legt eine neue eigenständige Textdatei (Inhaltsverzeichnis) auf Diskette ab.
Diese ist dann mit üblichen TP-Kommandos in die Ausgangsdatei an die gewünschte Stelle einzumischen (μKR).
Zur Erkennung einer Notation (Abschnittsnumerierung) gelten:
– die allgemeinen Vorschriften lt. TGL 0-1421
– beliebige Steuerzeichen des TP werden ignoriert
– Abschnittsnummer muß am Anfang einer Zeile stehen
– die maximale Länge der Notation beträgt 16
– die höchste Abschnittsnummer kann 999 sein (in jeder Stufe).
Am Display ist während der Erstellung des Inhaltsverzeichnisses eine Statistik über Verlauf und Stand der Verarbeitung ersichtlich. So werden Überschriftszeilen angezeigt, der zu verarbeitende Text ist wahlweise protokollierbar, jeweils drei Zeilen im Rollmodus. Außerdem erfolgt die Kontrolle auf aufsteigende Gliederungspunkte (-stufen). Die den Überschriften genügenden Konventionen (was gilt als solche) sind ebenfalls abrufbar. Von den TP-Kommandos gelangt logischerweise nur das DOT-Kommando .PNn in die Verarbeitung zur Fortschreibung der Seiten-Nummer. Eine Dokumentation liegt als TP-Datei (3 Seiten) vor.

VEB Datenverarbeitungszentrum Cottbus, Wilhelm-Pieck-Str. 5, Cottbus, 7500; Tel. 59 23 11

Böhm



Entwicklungen und Tendenzen

Zuwachs bei der PS/2-Familie

Die Lücke zwischen den 80286-Modellen 50 und 60 des PS/2 und dem 80386-Spitzenmodell 80 hat IBM mit der im Juni erfolgten Vorstellung des 80386-Modells 70 nunmehr geschlossen. Damit bestätigten sich Informationen, daß IBM der 80386-CPU gegenüber der 80286-CPU unter OS/2 und „Extended Edition" den Vorzug geben möchte. Das Modell 70 ist in einem kompakten Gehäuse als Auftischgerät untergebracht, so wie es Compaq und andere 80386-PC-Hersteller seit längerem praktizieren. Die äußere Form entspricht der des Modells 50. Es gibt Prozessorvarianten mit 16, 20 oder als Spitzenversion mit 25 MHz, womit die Geschwindigkeit des Konkurrenten Compaq 386/20 um bis zu 25 Prozent übertroffen werden soll. Allerdings konterte Compaq bereits drei Wochen später mit der Vorstellung seines 25-MHz-PC Deskpro 386/25.
Die PS/2-Reihe wurde weiterhin um das Modell 25 und das Modell 50 Z ergänzt. Wie sich damit nun die PS/2-Familie darstellt, zeigt die Tafel.

MP

	Modell 25	Modell 30	Modell 50	Modell 50Z	Modell 60	Modell 70	Modell 80
Prozessor	8086	8086	80286	80286	80286	80386	80386
Standard-Hauptspeicher	512/640 KByte	640 KByte	1 MByte	1 MByte	1 MByte	2 MByte	2 MByte
Maximal-Hauptspeicher	640 KByte	640 KByte	16 MByte	16 MByte	16 MByte	16 MByte	16 MByte
Festplatten-kapazität	–	–	20	20	30/60	44, 70, 115, 120	44, 70, 115, 314
Erweiterungs-steckplätze	2	3	3	3	7	3	7
Betriebssystem	DOS 3.3	DOS 3.3	DOS 3.3, OS/2	DOS 3.3, OS/2	DOS 3.3, OS/2	DOS 3.3, OS/2, AIX PS/2	DOS 3.3, OS/2, AIX PS/2

ICs mit 0,2 μm Linienbreite

Die Voraussetzung zur Massenproduktion von ICs, die auf einer Linienbreite von 0,2 μ beruhen, wurde von der japanischen Firma NEC mit der Entwicklung einer Röntgenstrahl-Lithographie-Anlage geschaffen. Gegenüber bekannten Lösungen bestehen die Neuerungen darin, daß die Röntgenstrahlen parallel bleiben und so mehrere Flächen der späteren Chips gleichzeitig bearbeitet werden können. Mit der Anlage sollen vierzig 8-Zoll-Wafer pro Stunde mit Flächen von 25 × 25 mm² bearbeitet werden können.

Quelle: Elektronik. – München 37 (1988) 1. – S. 7

Wi

Optikplatten

Eine Platte für optische Aufzeichnung hat die Firma Verbatim Corp. entwickelt. Durch die Möglichkeit, die Daten zu löschen, wird die Optikplatte zum Konkurrenten für Magnetplatten. Das Löschen wird durch punktförmiges Erwärmen mittels Laserstrahl und gleichzeitiges Drehen der Schichtpartikel durch ein angelegtes Magnetfeld erzielt.
Die Platte weist einen Durchmesser von 8,75 cm auf. Die Speicherkapazität beträgt 50 MByte, die Zugriffszeit 70 ms, die Transferrate 1 MByte/s.
Optische Speicher mit nur einmal beschreibbarem Datenträger werden bereits von verschiedenen Firmen angeboten. Hitachi stellte das Modell CDR 3500 vor, das sich durch die flache Bauweise anstelle eines zweiten Diskettenlaufwerkes in Personalcomputern einsetzen lasse.
Löschbare optische Platten werden den Einsatz optischer Speicher wesentlich beschleunigen.

Quelle: Elektronik. – München 37 (1988) 1. – S. 7, 30

Wi

Piezoelektrisches Kühlelement

Zur Kühlung einzelner Bauelemente oder Baugruppen wurde von der Firma Piezo Electric Products Inc. ein Bauelement entwickelt, das die Firma als „piezoelektronischen Gebläse-Kühler" bezeichnet. Die Gebläsewirkung wird durch ein flaches Element aus piezoelektrischem Material erzeugt, an welches eine Spannung angelegt wird und das durch Schwingungen die angrenzende Luftschicht bewegt. Ein derartiges Bauelement kann ein Luftvolumen von 1,5 cfm (cubic feet per minute) an dem zu kühlenden Bauelement vorbeibewegen.
Der Vorteil dieses Bauelementes liegt in den geringen Abmessungen (Grundfläche 38 × 44 mm², Dicke 5 mm). Die Leistungsaufnahme beträgt 170 mW.

Quelle: Elektronik. – München 37 (1988) 2. – S. 7

Fa

Rechner Cray-3 mit Galliumarsenidschaltkreisen

Nachdem die Firma Cray für das dritte Quartal 1988 den Rechner Y-MP/832 ankündigte, der sich durch eine dreifach höhere Leistung gegenüber dem Vorgängermodell X-MP auszeichnen soll, wurde nunmehr mit dem Rechner Cray-3 ein weiterer Superrechner angekündigt, dessen Leistung nochmals um den Faktor 3 gesteigert wurde. Während beim Rechner Y-MP die Leistungssteigerung durch die Parallelverarbeitungsarchitektur von 8 Prozessoren erreicht wurde, kommen beim Cray-3 Hochgeschwindigkeitschips aus Galliumarsenid zum Einsatz. Offensichtlich will die Firma Cray ihre beiden Superrechnerlinien fortführen, denn für den Zeitraum 1992 bis 1993 wurde der Rechner Cray-4 und für 1991 ein Nachfolger des Y-MP angekündigt.

Quelle: Blick durch die Wirtschaft vom 17. 2. 1988

Wi

Superkurzer Laserimpuls

Von einer Forschergruppe der schottischen Universität in Fill wurde ein Laser entwickelt, der 19 fs (Femtosekunden) Impulsdauer aufweist. Damit wurde eine wesentliche Verkürzung gegenüber den bisher erreichten Werten erzielt. Kurze Impulse sind für Laseruntersuchungen, Halbleiter, optoelektronische Geräte und in der Prozeßtechnik Voraussetzung für weitere Fortschritte.

Quelle: Elektronik. – München 37 (1988) 1. – S. 7

Fa

Verbrauch von Druck- und Schreibpapier steigt weiter an

Trotz der weiteren Verbreitung elektronischer Kommunikationsmittel (Mailbox, Elextronic Mail) wird von der ECC International Ltd. in einer Studie für den Zeitraum 1986 bis 1991 ein Zuwachs des Verbrauchs von Druck- und Schreibpapier um jährlich 3 Prozent vorausgesagt.

Quelle: BTS. – Aachen 33 (1987) 12. – S. 9

Fa

Textprozessor mit geringen Abmessungen

Die Firma Sharp stellte einen portablen Textprozessor für die englische und japanische Sprache vor. Der Prozessor verfügt über ein 2-Zoll-Plattenlaufwerk und einen Thermodrukker mit einer Auflösung von 25 × 25 Punkten. Sonderzeichen der deutschen, französischen und spanischen Sprache können durch den Drucker dargestellt werden. Für Texte in Englisch wird eine Rechtschreibkontrolle angeboten.

Quelle: Elektronik. – München 37 (1988) 2. – S. 7

Fa

Chipsatz für Grafiksysteme

Einen Chipsatz in modularer Bauweise zur Anpassung an die jeweiligen Anforderungen hinsichtlich Auflösung und Farbvielfalt wurde von der Firma National Semiconductor entwickelt. Das Schlüsselbauelement ist ein programmierbarer Grafikprozessor mit integrierter Beschleunigerhardware. Die Grafikfunktionen werden in mehreren Bausteinen in der dafür jeweils optimalen Technologie realisiert. Die offene Architektur soll die Anpassung an unterschiedliche Hard- und Softwareumgebung erleichtern.
Der Chipsatz umfaßt Video-Taktgeneratoren für hohe Bildpunktfrequenzen (bis 225 MHz), Video-Schieberegister, Steuerbausteine für Bildspeicher und einen Rasterprozessor zur Entlastung der CPU. Den Systementwicklern sollen Softwarewerkzeuge auf Assembler-, C- und Grafikstandardebene zur Verfügung stehen.

Quelle: eee. – Leinfelden-Echterdingen (1988) 3. – S. 16

Wi

Lokale Netze auf Glasfaserbasis

Gegenwärtig verlangen die verschiedenen Adapterkarten für Netzwerke spezielle Kabelsysteme, die untereinander nicht kompatibel sind. Mit der Verkabelung ist somit bereits entschieden, welche LAN-Kategorie eingesetzt werden kann. Soll die Leistung des LAN wachsenden Forderungen angepaßt werden, ist meist eine Neuverkabelung notwendig. Glasfasern gestatten Transferraten, die den Ansprüchen der nächsten Jahre gerecht werden und auch die Forderungen von Bild- und Tonübertragung berücksichtigen. Neben der hohen Leistung besitzen Glasfasern eine Reihe von Vorteilen bei der Verlegung, die höhere Kosten rechtfertigen. So können Glasfasern in vorhandenen Kabelschächten verlegt werden, da Reflexionen und magnetische und elektrische Störungen wirkungslos sind.
Das auf Glasfasertechnologie basierende LAN FX 4400 der Firma Fibermux Corp. gestattet, 256 Netze zu einem Netzwerk in Punkt-zu-Punkt-, Stern- oder Ringtopologie bis zu 20 km entfernt zusammenzuschalten. Das modulare System gestattet 384 Anschlüsse für Sprachübertragung, 16 Anschlüsse mit T1-Interface, 16 Anschlüsse X.21-Interface. Ein Netzwerk kann aus bis zu 9 FX-4400-Systemen bestehen.

Quelle: eee. – Leinfelden-Echterdingen (1988) 1/2. – S. 28

Wi

Elektronikanteil im Automobilbau steigt weiter an

Auf einem Kolloquim von Fachleuten des Automobilbaus und der Elektronik in Köln wurde eingeschätzt, daß gegenwärtig 1/6 der Kosten im Autobau auf Elektronik und Elektrik entfallen. Bis zum Jahr 2000 soll sich dieser Anteil auf 1/3 erhöht haben.
Die Modelle des Jahres 1995 sollen u. a. über 1000 Chips verfügen, die die von 100 Sensoren aufgenommenen Informationen auswerten und an 80 Aktoren weiterleiten. Die Fahrzeuge sollen mit etwa 45 kleinen Motoren ausgerüstet sein. Fünf Displays übermitteln dem Fahrer die entsprechenden Informationen über Fahrzeug, Fahrweg u. ä.

Quelle: eee. – Leinfelden-Echterdingen (1988) 1/2. – S. 12

Wi

Sozialistische ökonomische Integration

ZEITSCHRIFTEN

aus RGW-Ländern gehören dazu!



aus der Sowjetunion

Mikroehlektronika
Mikroelektronik
in russischer Sprache
Index-Nr.: Z 70571
Abonnementspreis: 12,30 M
für 6 Monate
Erscheint sechsmal jährlich

Mikroprocessornye sredstva i sistemy
Mikroprozessoren, Mittel und Systeme
in russischer Sprache
Index-Nr.: Z 70588
Abonnementspreis: 23,10 M für 1 Jahr
Erscheint sechsmal jährlich

Programmirovanie
Programmierung
in russischer Sprache
Index-Nr.: Z 70763
Abonnementspreis: 33,60 M für 1 Jahr
Erscheint sechsmal jährlich

Programmnoe obespechenie
Software
in russischer Sprache
Index-Nr.: Z 55461
Abonnementspreis: 192,– M für 1 Jahr
Erscheint einmal monatlich

aus der VR Bulgarien

Computer za vas
Computer für Euch
in bulgarischer Sprache
Index-Nr.: Z 20593
Abonnementspreis: 14,10 M
für 3 Monate
Erscheint einmal monatlich

Elektronizacija i robotika
Elektronisierung und Robotertechnik
in bulgarischer Sprache
Index-Nr.: Z 21575
Abonnementspreis: 21,75 M
für 6 Monate
Erscheint sechsmal jährlich

aus der VR Polen

Przeglad Dokumentacyjny – Nauki i Techniki Komputerowe
Dokumentationsrundschau – Computerwissenschaft und -technik
in polnischer Sprache
Index-Nr.: Z 37183
Abonnementspreis: 97,20 M
für 6 Monate
Erscheint sechsmal jährlich

aus der ČSSR

Computers and artificial intelligence
Computer und künstliche Intelligenz
in englischer und russischer Sprache
Index-Nr.: Z 49484
Abonnementspreis: 38,70 M
für 6 Monate
Erscheint sechsmal jährlich

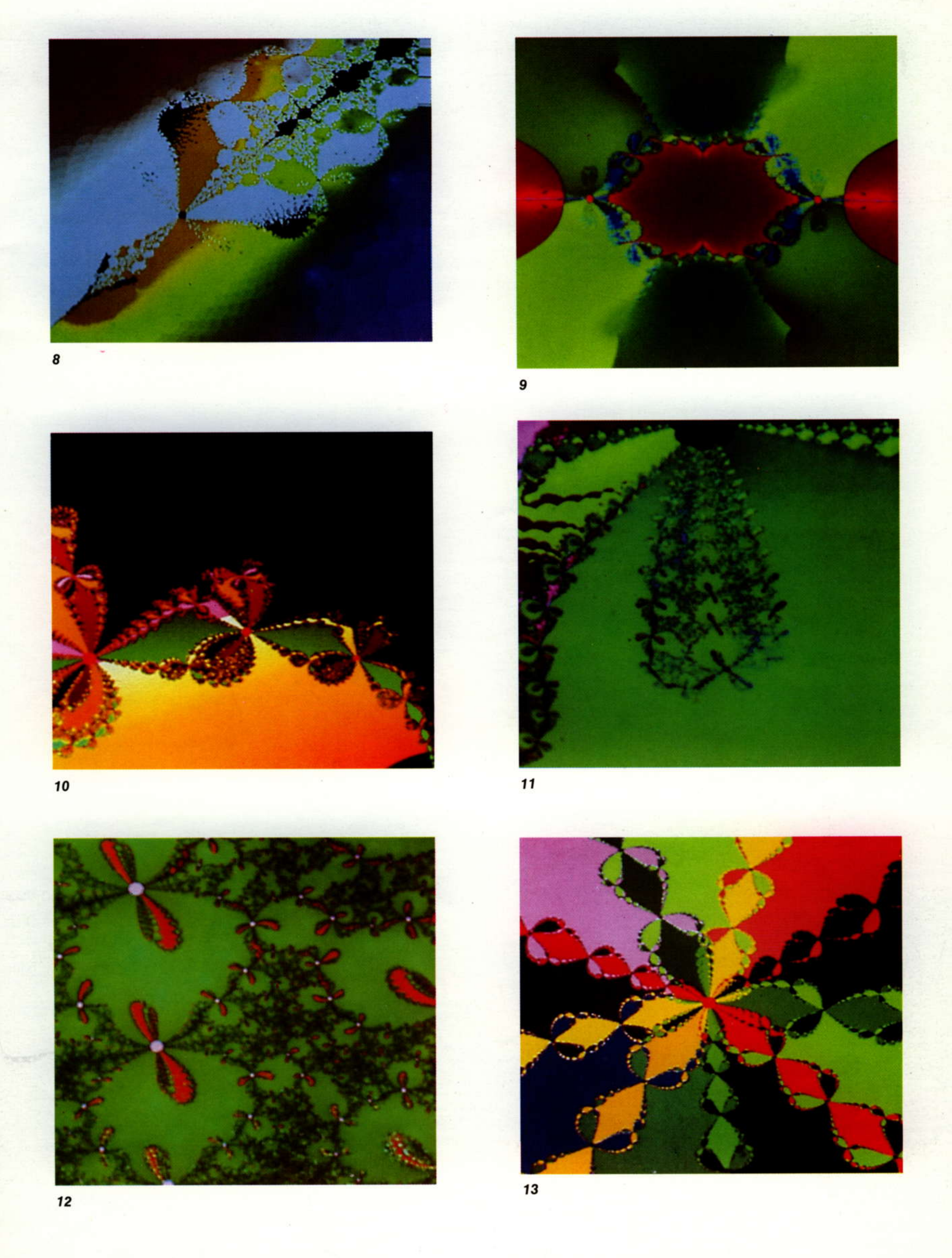
aus der Ungarischen VR

Computerworld Szamitastechnika
Computerwelt – Rechentechnik
in ungarischer Sprache
Index-Nr.: Z 0237-7837
Abonnementspreis: 66,70 M für 1 Jahr
Erscheint dreiundzwanzigmal jährlich

Információ – Elektronika
Informationselektronik
in ungarischer Sprache
Index-Nr.: Z 0019-9753
Abonnementspreis: 34,50 M
für 6 Monate
Erscheint sechsmal jährlich

Ihr Postzeitungsvertrieb informiert Sie über die Bestelltermine für diese Zeitschriften und über weitere in der Postzeitungsliste, Teil II – Ausland –, enthaltene Zeitschriften dieser Fachgebiete.

DEUTSCHE POST – POSTZEITUNGSVERTRIEB



8

9

10

11

12

13

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 9
S. 257–288

LCD Einheit 40
CRT/FDD Einheit

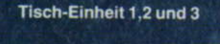
LCD Einheit 80 und 80/2

Cartridges

Basis-Modul

Touch Keyboard

Schreibmaschinen Keyboard



Tisch-Einheit 1,2 und 3



Viele Kombinationsmöglichkeiten

CPU V20 (8088-kompatibel), 4,77/10 MHz umschaltbar, Arbeitsspeicher 256 KB bis 640 KB, RAM-Disk (Option) bis 768 KB, System-ROM mit MS-DOS 3.2 bis 128 KB, CMOS EPROM, bis 512 KB CMOS mask ROM, Echtzeit-Uhr, Piezo-Lautsprecher, Haupt-Batterie 2000 mAh, Sub-Batterie 500 mAh

LCD 80, 80/2
80 Zeichen, 25 Zeilen, 640 × 200 Punkte, STN, Voll-Reflexion, LCD 80/2: wie LCD 80 mit CRT- und FDD-Schnittstellen

LCD 40
40 Zeichen, 10 Zeilen, 320 × 80 Punkte, STN, Voll-Reflexion

Anwendungs-Cartridges
Printer-Cartridge (40 Zeichen/Zeile), Universal-Cartridge, anwenderspezifisch

Touch Keybord
Touchkeys: 96 (8 Zeilen × 12 Spalten Text), LCD: 200 × 88 Punkte (11 Zeilen × 25 Zeichen)

Schreibmaschinen-Tastatur
79 Tasten, mit Cursor- und Funktionstasten, LED-Anzeigen

Tischeinheiten
Batterie und 1 FDD 3 1/2" (720 KB) oder 2 FDD 3 1/2" (720 KB) oder 1 FDD 3 1/2" (720 KB) und 3 1/2" HDD (20 MB)

Sonstige Optionen
Batteriegepufferte RAM-Boards 384 KB, FDD TF-16: 2 × 5 1/4" (360 KB), Ladegerät 5 V, 4 A für 100–240 V, Boards mit seriellen Schnittstellen, Barcode-Lesestift

Mitgelieferte Software
MS-DOS 3.20 und spezielle Dienstprogramme

Maße/Gewichte
315 (B) × 224 (T) × 47 (H) mm³/ca. 3 kg

LCD Einheit 40. Zur Wiedergabe von Daten und Kurztexten.
CRT/FDD Einheit. Zum Anschluß eines Color-Monitors und der Disketten-Station TF-16.
LCD Einheit 80 und 80/2. Zur Wiedergabe von Texten, Tabellen und Grafiken. LCD 80/2 mit Schnittstellen für Color-Monitor und Diskettenlaufwerk TF-16.
Basis-Modul. Hauptplatine mit 8088-kompatiblem, aber schnellerem Prozessor (V 20). Batteriegepufferte System-, Arbeitsspeicher-(max. 640 KB) und RAM-Disk (max. 768 KB). System-ROM mit MS-DOS 3.2, 3 Steckplätze für Applikations-ROMs. Dazu Modul-Anschlüsse, serielle und parallele Schnittstellen und Anschluß für Barcode-Lesestift.
Anwendungs-Cartridges. Für Drucker, RAM- und ROM-Erweiterungen (teilweise von PX-4 zu übernehmen).
Touch Keyboard. Links: Großes Touch-Panel: LCD-Anzeige und freigestaltbare Eingabe-Einheit in einem. Ideal für anwenderspezifische Lösungen. Rechts: Numerischer Tastenblock.
Schreibmaschinen Keybord. Mit 79 Tasten für 101 bzw. 102 Funktionen und LED-Anzeigen für Num Lock, Caps Lock, Scroll Lock.
Tisch-Einheit 1,2 und 3. Mit eingebauter Batterie, 3 1/2" Diskettenlaufwerk oder zusätzlich zweitem Diskettenlaufwerk oder 3 1/2" Hard-Disk mit 20 MB.
Viele Kombinationsmöglichkeiten. Je nach Anwendung lassen sich verschiedene PCs zusammenstellen, die sich zusätzlich individuell ausrüsten lassen.

Computer in der Hand

Schon bald nach dem ersten massenhaften Einsatz von integrierten Schaltungen in der Computertechnik – insbesondere Mikroprozessoren – sahen Hersteller Möglichkeiten zur Entwicklung einer neuen Rechnerklasse – der tragbaren Computer. Der Begriff „tragbar" ist nun sehr dehnbar, und demzufolge lassen sich darunter echte universelle Handcomputer (Hand-Helds) um die 3 kg Gewicht, sogenannte Schoßcomputer (Laptops) um die 6 kg, aber auch Portables um die 10 kg einordnen. Letztere stellen mit ihrer Leistungsfähigkeit als „PC am Griff" oft schon Desk-Top-Computer (Auf-Tisch-Modelle) in den Schatten.
Die Gemeinsamkeit von Hand-Held-Computern und Laptops besteht vor allem in der Grundfläche, die etwa einem A4-Blatt entspricht und in dem klappbaren Display, die Gemeinsamkeit von Laptops und Portables darin, daß sie meist keine Aufgaben wirklich mobiler Datenverarbeitung erfüllen, sondern im Sinne einer durchgängigen DV-Organisation stationäre Lösungen transportieren. Sie entsprechen in ihrer Grundkonfiguration deshalb meist Desk-Tops und unterscheiden sich von diesen nur durch geringeres Gewicht und kleinere Abmessungen.
Der erste Hand-Held-Computer war der HX-20 von Epson – noch mit einem sehr kleinen Display ausgestattet. Es folgten die Generationen mit 8-Bit-CMOS-Prozessoren, mit der Integration des Betriebssystems CP/M bis zur Ausstattung mit 16-Bit-Prozessoren und MS-DOS.
Mit dem auf der Leipziger Messe gezeigten und hier vorgestellten Hand-Held PX-16 will Epson nun seinen Vorstellungen von der mobilen Datenverarbeitung Rechnung tragen; Hand-Helds sollen Alleskönner sein: im Flugzeug zur Bordfunkunterstützung, im Versuchsauto als Steuercomputer für die Meßinstrumente, im Rettungswagen zur exakten Diagnose, im Kundendienst als Handlagerverwalter usw. – das heißt, weit über die Aufgaben der klassischen Betriebsdatenerfassung hinausgehend. Um der jeweiligen Aufgabe optimal entsprechen zu können, wurde ein modularer Aufbau für Hand-Held-Computer entwickelt, der eine hohe Flexibilität gewährleistet.
Auch wurde die Vielzahl von Eingabemöglichkeiten – Sensor, Thermofühler, Barcode-Lesestift, IC-Karte – und Ausgabemöglichkeiten – Drukker, RAM-Karte, Modem zur Kommunikation – berücksichtigt.
Mit den auswechselbaren Modulen kann der PX-16 also für die genannten Aufgaben jeweils zu einem „maßgeschneiderten" Computer kombiniert werden, als Laptop beispielsweise mit Schreibmaschinentastatur, 80 × 25-Zeichen-Vollbildschirm und Floppy- oder Hard-Disk. *MP-We*

Grafik: Epson; Foto: Weiß

ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK



Heft 10 1988

Mikroprozessortechnik

VEB Verlag Technik Berlin
ISSN 0232-2892

MODULA-2: PASCAL ohne die Nachteile von PASCAL

dBASE III im Vergleich

MC 68010 im Überblick

Bildungscomputer A 5105

Videoton VT 180 (Ungarn)

Als Weiterentwicklung des PC/AT-kompatiblen VT 160 stellte die ungarische Firma Videoton während der diesjährigen Budapester Messe (über die wir Sie am Schluß dieses Heftes informieren) ihren ersten PC mit 32-Bit-Prozessor vor, den VT 180. Aufgrund seiner AT-Kompatibilität laufen auf ihm alle Programme, die auf XT-kompatiblen PCs (beispielsweise Videoton VT 110) und AT-kompatiblen PCs erstellt wurden.
In das Gehäuse können bis zu 4 Massenspeicher integriert werden, ansteuerbar über eine gemeinsame Controllerkarte. Dies sind zum Beispiel max. 2 Floppy-Disk-Laufwerke (5,25 Zoll), 2 Festplattenlaufwerke in slim-line-Ausführung oder ein Streamer.
Die Kapazitäten betragen bei den Floppies 360 KByte oder 1,2 MByte, bei den Festplatten 20, 40 oder 80 MByte und beim Streamer 60 MByte.
Auf der Systemplatine befinden sich der 32-Bit-Prozessor, getaktet mit 6 oder 16 MHz, optional ein 80387-Koprozessor, 2 MByte RAM (aufrüstbar auf 16 MByte), ein 32 Bit breiter Speicherbus und ein Shadow-RAM für das ROM-BIOS. Außerdem sind auf der Systemplatine 5 Steckplätze für 16-Bit-Erweiterungskarten und 4 Steckplätze für 8-Bit-Erweiterungskarten.



Foto: Weiß

Als Bildschirm können wahlweise Farbgrafik- oder Monochrom-Monitore verwendet werden.
Die Hercules-kompatible MDA-Karte ermöglicht die Ansteuerung von 720 × 348 Bildpunkten bzw. 80 × 25 alphanumerischen Zeichen.
Die 256 Zeichen sind in einem 4-KByte-ROM abgelegt. Auf der Karte befindet sich außerdem ein Centronics-Interface.
Der Farbgrafikadapter (CGA) enthält den Controller und 16 KByte Bildspeicher. Er erlaubt folgende Grafikmodi:

- 640 × 200 Bildpunkte mit 2 Farben
- 320 × 200 Bildpunkte mit 4 Farben
- 160 × 100 Bildpunkte mit 16 Farben.

Die 256 alphanumerischen Zeichen sind in 25 Zeilen zu je 80 Zeichen darstellbar. Auf der Karte befinden sich ein Video- und ein RGB-Ausgang, ein Lichtstift-Eingang sowie ein Centronics-Interface.
Die hochauflösende Farbgrafikkarte (EGA) beinhaltet neben dem Controller 256 KByte RAM. Es sind zwei wählbare Grafikmodi mit je zwei Formaten möglich:

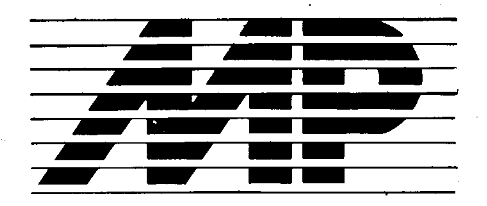
- 640 × 350 oder 720 × 348 Bildpunkte mit zwei Farben
- 640 × 350 oder 640 × 200 Bildpunkte mit 16 aus 64 Farben.

Diese Karte hat neben dem Zeichengenerator einen Video- und RGB-Ausgang sowie einen Lichtstift-Eingang.
Für den VT 180 stehen zwei Tastaturen zur Auswahl: entweder die englische Tastatur mit 84 Tasten oder eine ungarische Tastatur mit 113 Tasten.
Zur Erweiterung der Einsatzmöglichkeiten des VT 180 – beispielsweise als File-Server in Netzen – gibt es im Lieferprogramm verschiedene zusätzliche Interfaces.
Die Software umfaßt unter anderem das Videoton-MS-DOS 3.2 und GWBASIC.

MP



Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 10 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR–1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2870371); Herbert Hemke, Redakteur (Tel. 2870203); Sekretariat Tel. 2870381

Gestaltung Christina Bauer

Titelfoto Herbert Hemke

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

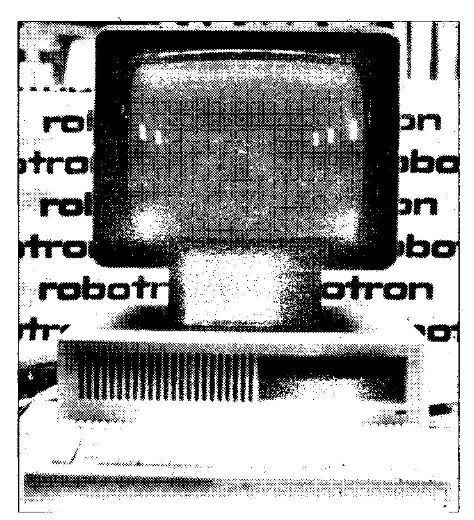
Redaktionsschluß: 16. August 1988

AN (EDV) 49837

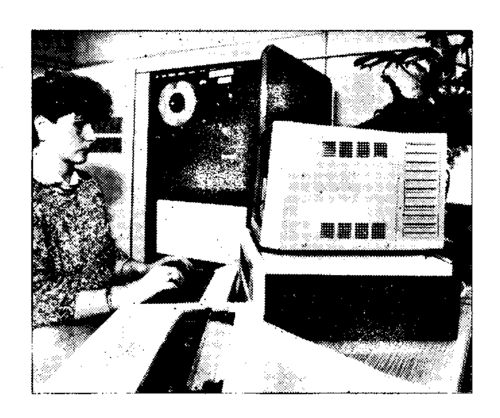
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tisku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Há Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



Nachdem wir in MP 7/88 bereits kurz über den neuen Bildungscomputer von Robotron informierten, stellen wir ihn auf der **Seite 292** sowie auf der **4. Umschlagseite** näher vor. Darüber hinaus vergleichen wir ihn mit den bisher verfügbaren Kleincomputern aus unserer Produktion.



In unserer Rubrik *vorgestellt* machen wir Sie auf der **Seite 311** mit dem *Rechnersystem mit virtuellem Speicher* K 1840 von Robotron bekannt. Es ermöglicht u. a. die parallele Arbeit von bis zu 16 Nutzern.

Vorschau
In Heft 11 finden Sie Beiträge zu folgenden Themen:
- ASIC – eine Revolution?
- Rekursion – ein Baum voller Menüs
- EPROMs hoher Speicherkapazität

Inhalt

Mikroelektronikindustrie entwickelt sich erfolgreich

Der VEB Mikroelektronik „Karl Marx" Erfurt, Stammbetrieb des Kombinates Mikroelektronik, ist größter Produzent von unipolaren Festkörperschaltkreisen in der DDR. 1968 war in Erfurt mit dem Beginn der Diodenfertigung die Grundlage der elektronischen Industrie geschaffen worden, erläuterte Prof. Dr. Heinz Wedler, Generaldirektor des Kombinats Mikroelektronik. 1977 erfolgte mit der Herstellung des ersten Mikroprozessortyps U 808 der Einstieg in die Mikroelektronik. Ein Jahr später erhielt die Mikroelektronik der DDR durch die Gründung eines eigenen modernen Industriekombinates in Erfurt eine einheitliche materielle und organisatorische Basis.
Dem VEB Mikroelektronik „Karl Marx" wurde die Aufgabe gestellt, die Volkswirtschaft bedarfsgerecht mit unipolaren Schaltkreisen zu versorgen. Das erforderte, am Stadtrand von Erfurt-Südost neue Produktionsanlagen zu schaffen, von denen bereits zwei Chipfabriken produzieren. Gegenwärtig entsteht parallel zu den Produktionsanlagen ein Forschungszentrum für unipolare hoch- und höchstintegrierte Schaltkreise, teilte Prof. Wedler mit.
1986 nahm eine Spezialschule für mathematisch-naturwissenschaftlich begabte Schüler als „Kaderschmiede" künftiger Spezialisten den Lehrbetrieb auf. Als nächstes kommen ein polytechnisches Zentrum, eine Betriebsschule und ein Technikum hinzu. Alle im Kombinat Mikroelektronik tätigen Wissenschaftler, Ingenieure und Facharbeiter nehmen an der Weiterbildung teil.

ADN

Rechentechnische Basis verstärkt

Die rechentechnische Basis in der DDR ist mit der Produktion von 27 241 Büro-, Personal- und Arbeitsplatzcomputern ausgebaut worden. Das ergibt sich aus der Mitteilung der Staatlichen Zentralverwaltung für Statistik über die Durchführung des Volkswirtschaftsplanes im ersten Halbjahr 1988. Wesentlich gesteigert wurde die Herstellung von 32-Bit-Computern sowie 16-Bit-Computern mit größerer Rechengeschwindigkeit und Speicherkapazität.
Gegenwärtig sind in der Volkswirtschaft 57 000 CAD/CAM-Arbeitsstationen und -systeme im Einsatz, mit deren Nutzung ein ständig wachsender Beitrag zur Steigerung der Arbeitsproduktivität und zur Erhöhung der Effektivität geleistet wird. Zur Sicherung der ökonomischen Wirksamkeit der CAD/CAM-Technik trug entscheidend das Wachstum der Softwareherstellung um 45 Prozent bei.
Die Anzahl der eingesetzten Industrieroboter wuchs auf insgesamt 84 400. Durch ihren effektiven Einsatz wurden die Rationalisierung und Automatisierung der Produktion weiter beschleunigt.

ADN

Massenproduktion von 64-KBit-RAMs

Die mikroelektronische Basis der DDR-Volkswirtschaft ist im ersten Halbjahr 1988 mit der Aufnahme der Großserienproduktion von 64-KBit-Speicherschaltkreisen im Forschungszentrum Mikroelektronik Dresden und im VEB Mikroelektronik „Karl Marx" Erfurt ausgebaut worden. Damit wird es möglich, massenhaft Schaltkreise für ein höheres Niveau der Automatisierungs- und CAD/CAM-Technik zu nutzen.
Vier Millionen Stück, 1,5 Millionen mehr als ursprünglich geplant, sollen in diesem Jahr den Anwendern zur Verfügung gestellt werden.
Zudem wurden weitere technologische Spezialausrüstungen für die Mikroelektronik entwickelt und in die Produktion überführt. Das trug dazu bei, gegenüber dem ersten Halbjahr 1987 insgesamt 20 Prozent mehr Spezialausrüstungen für die Herstellung aktiver Halbleiterbauelemente bereitzustellen. Im gleichen Zeitraum wuchs die Produktion von monolithisch-integrierten Schaltkreisen um 44 Prozent und von mikrolithografischen Geräten um 23 Prozent. Den Grundstein für diese Leistungen legten die Kollektive der Kombinate Carl Zeiss JENA und Mikroelektronik Erfurt mit einer um 12 beziehungsweise 21 Prozent gewachsenen Produktivität.

ADN

EC 1834 in Serienproduktion

Im Büromaschinenwerk „Ernst Thälmann" Sömmerda begann jetzt die Serienproduktion des neuen 16-Bit-Computers EC 1834. An der Entwicklung des Rechners, der im Vergleich zum PC 1715 die zwei- bis 2,5fache Speicherkapazität und Verarbeitungsgeschwindigkeit besitzt, waren mehrere Betriebe des Kombinats Robotron sowie Partner aus anderen Einrichtungen und Betrieben der Elektrotechnik/Elektronik beteiligt. Für die Serienproduktion des neuen Personalcomputers, der das Gütezeichen „Q" trägt, wurden in Sömmerda die technologischen Voraussetzungen parallel zur Erzeugnisentwicklung geschaffen. Dazu gehörte die Erarbeitung spezieller Software für elektronische Prüf- und Meßtechnik. Die Kollektive des Büromaschinenwerkes Sömmerda stellen in diesem Jahr mehr als 28 000 Personalcomputer, davon 5000 des neuen Typs, her. Der EC 1834 wird außerdem im Buchungsmaschinenwerk Karl-Marx-Stadt produziert.

ADN

Cottbuser Softwarebörse

Auf der ersten Softwarebörse des Bezirkes stellten im Juni in Cottbus 25 Betriebe über 50 Programmlösungen für 8-Bit-Rechner vor. Die Veranstalter – Datenverarbeitungszentrum, Bezirksvorstand der Kammer der Technik und Neuererzentrum des Bezirkes – hatten Programme für Leitung, Planung, Rechnungsführung und Statistik ausgewählt, die von Fachleuten an zehn Personalcomputern demonstriert wurden. So offerierte das Braunkohlenwerk „Glückauf" Knappenrode Beispiele zur Vorbereitung von Investitionsobjekten, zur wissenschaftlichen Arbeitsorganisation und zur Normenstatistik. Andere Teilnehmer, darunter Kleinbetriebe, waren mit rechnergestützter Kontrolle der Neuererarbeit, des Energie- und Materialverbrauchs vertreten. Am Bildschirm konnten sich die Besucher außerdem über rund 650 nachnutzbare Lösungen der Software-Referatebank des Bezirkes informieren. Programme zur technischen Vorbereitung der Produktion bietet eine der nächsten Softwarebörsen an, die jeweils für ein spezielles Einsatzgebiet von Computern ausgerichtet werden.

ADN

DDR-Minister bei bulgarischem Ministerratsvorsitzenden

Der Vorsitzende des Ministerrats der Volksrepublik Bulgarien, Georgi Atanassow, empfing im Juni in Sofia den Minister für Elektrotechnik und Elektronik der DDR, Felix Meier, zu einem Gespräch. Dabei befürworteten beide Seiten die Vertiefung der Zusammenarbeit auf dem Gebiet der Rechentechnik, der Mikroelektronik und des Elektronikmaschinenbaus. Während des Arbeitsbesuches hatte Felix Meier mit dem Vorsitzenden der Wirtschaftsvereinigung „Elektronika", Minister Iwan Tenew, die Erweiterung der Zusammenarbeit in verschiedenen Bereichen der Schlüsseltechnologien sowie aktuelle Fragen der Wirtschaftskooperation erörtert. Er besichtigte mehrere Betriebe der elektronischen Industrie in Stara Sagora, Gabrowo, Prawez und Sofia.

ADN

Robotron-Doppelrechnersystem an Versorgungszentrum Moskau

Ein Doppelrechnersystem EC 1055.M wurde im Juni durch ein Inbetriebnahmekollektiv des VEB Robotron-Anlagenbau in Leipzig an das Versorgungszentrum Moskau übergeben. Die Inbetriebnahme des EC 1055.M stellt die erste Etappe eines umfangreichen Vorhabens in der sowjetischen Metropole dar. Bis zum vierten Quartal dieses Jahres wird ein weiteres solches System montiert und beide Datenverarbeitungsanlagen zu einem Vierrechnerkomplex gekoppelt. Damit erschließen sich dem sowjetischen Anwender erweiterte Möglichkeiten, Versorgungsaufgaben mit Hilfe der Datenverarbeitung effektiv zu lösen. Der EC 1055.M gehört zu den EDV-Anlagen der mittleren Leistungsklasse des ESER und verfügt über eine Rechengeschwindigkeit von 450 000 Operationen in der Sekunde.

ADN

Konsultationspunkt für Software bei Robotron

Im Betrieb Robotron-Elektronik Dresden wurde im Juni der erste bezirkliche KDT-Konsultationspunkt „Software" eröffnet. Mit dieser Einrichtung wollen Bezirksvorstand und Betriebssektion der sozialistischen Ingenieurorganisation dem steigenden Bedarf an Informationen und Dokumentationen zur Softwareproduktion, insbesondere für die dezentrale Rechentechnik, entsprechen. Vorgesehen sind Konsultationen und Fachvorträge. Wie zur Eröffnung hervorgehoben wurde, diene der künftige Erfahrungsaustausch zur effektiveren Softwarearbeit auch der Realisierung der für 1988 im Bezirk Dresden vorgesehenen 235 KDT-Objekte zum Einsatz von Rechentechnik.

ADN

Pekinger Haidian-Bezirk – erste Adresse Chinas für Hochtechnologie

Der Pekinger Haidian-Bezirk, früher vor allem wegen seiner kaiserlichen Parkanlagen und Gärten berühmt, ist heute eine der ersten Adressen in der Volksrepublik China, wenn es um Forschung und Produktion im Bereich Hochtechnologie geht. Mit mehr als 50 Universitäten, Hoch- und Fachschulen sowie 138 Forschungseinrichtungen ist der im Nordwesten der Hauptstadt gelegene Distrikt der mit Abstand intelligenzintensivste des Landes. Die Regierung investiert seit 1949 in seinen Aufbau und seine

In eigener Sache

Für eine weitere Redakteurstelle in unserer Zeitschrift suchen wir einen geeigneten Mitarbeiter bzw. eine Mitarbeiterin mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik.
Zu den Aufgaben gehören unter anderem das redaktionelle und fachliche Bearbeiten von Manuskripten, der Besuch und die Auswertung von Fachtagungen, -messen und -ausstellungen, die Zusammenarbeit mit Autoren und Gutachtern sowie ggf. das Testen und Beurteilen von Programmen, die der Redaktion zur Veröffentlichung eingereicht werden.
Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, rufen Sie uns unter Tel. 2 87 03 71 oder 2 87 02 03 an oder schreiben Sie an:
VEB Verlag Technik, Redaktion MP, Oranienburger Str. 13/14, Berlin, 1020

ständige Weiterentwicklung rund zehn Milliarden Yuan.
Angezogen von dieser geballten wissenschaftlichen Potenz, haben sich hier rund 150 Betriebe unterschiedlicher Eigentumsformen angesiedelt und Produktionsstätten sowie Verkaufseinrichtungen für Computer und andere elektronische Erzeugnisse aufgebaut. Ein Teil der elektronischen Erzeugnisse aus dem Haidian-Bezirk, die bisher 13 internationale und 23 nationale Auszeichnungen errangen, wird exportiert, unter anderem nach Japan und in die USA.
Die Betriebe arbeiten mit wirtschaftlicher Rechnungsführung und sind für Gewinn und Verlust eigenverantwortlich. Die meisten ihrer Beschäftigten erhalten leistungsorientierte Gehälter und sind befristet eingestellt. Dadurch wird die in vielen chinesischen Betrieben noch anzutreffende sogenannte Beschäftigung auf Lebenszeit ausgeschaltet, die zu einer teilweisen Überbesetzung geführt hat und von führenden Wirtschaftsexperten des Landes als Hemmnis für eine Erhöhung der Effektivität angesehen wird.
Die Regierung und die Stadt Peking haben kürzlich beschlossen, die bisher auf das Gebiet um die Zhongguancun-Straße beschränkte Hochtechnologiezone von Haidian auf 100 Quadratkilometer auszudehnen und die Ansiedlung neuer Betriebe zu unterstützen. Das beinhaltet beträchtliche Steuervergünstigungen, weitreichende Verfügungsrechte über Deviseneinnahmen sowie die Gewährung von Krediten zu Vorzugsbedingungen. Außerdem wurde den Firmen das Recht eingeräumt, Hochschulabsolventen sowie ausländische Experten selbst einzustellen.
Wie Yang Zhenhua von der Vereinigung für Wissenschaft und Technologie in Haidian der Presse mitteilte, sind bereits 400 Anträge auf Firmengründung in der erweiterten Technologiezone gestellt worden. Unterdessen findet das Pekinger Beispiel bereits Nachahmung. Die Behörden von Lanzhou, Hauptstadt der Provinz Gansu, haben angekündigt, daß sie nach den hauptstädtischen Erfahrungen ebenfalls eine Entwicklungszone für Hochtechnologie einrichten werden.

ADN-Anton

Elektronikprodukte – Malaysias größte Devisenquelle

Die junge Elektronikindustrie Malaysias hat die einst wichtigsten Devisenquellen des Landes, Erdöl, Kautschuk, Zinn, Palmöl und Holz, verdrängt und die Spitze in der Exportliste eingenommen. Erstmalig wurde diese positive Tendenz in der Jahresbilanz 1986 sichtbar. Als Ursache gaben Experten zunächst den Preisverfall für Rohstoffe auf dem Weltmarkt an. Doch die Steigerungsraten im vergangenen Jahr machten deutlich, daß die elektronische Industrie die Rohstoffausfuhren endgültig von der führenden Exportposition verdrängt hat.
1987 wurden 16,2 Prozent der Exporteinnahmen des Landes durch die elektronische Industrie realisiert und der Exportplan mit 44 Prozent überboten. Damit war Malaysia nach den USA und Japan drittgrößter Exporteur von Halbleitern im nichtsozialistischen Währungsgebiet.
Insgesamt 60 000 Werktätige, das heißt jeder fünfte in der Industrie Beschäftigte, arbeiten in einem der 26 Betriebe, die elektronische Bauteile herstellen.
Die malaysische Behörde für industrielle Entwicklung (MIDA) führt die erfolgreiche Entwicklung des Elektroniksektors auf den 1986 in Kraft getretenen und auf zehn Jahre befristeten Generalplan zur industriellen Entwicklung des Landes zurück. In diesem Plan sind sowohl Ausbildungsprogramme für Facharbeiter als auch beträchtliche finanzielle Vergünstigungen für ausländische Investoren festgelegt. Für 19 der 26 ausnahmslos in ausländischer Hand – zumeist Japans oder der USA – befindlichen Elektronikbetriebe sind die Vergünstigungen bereits erloschen. Aber kein Unternehmen hat sich bisher aus Malaysia zurückgezogen. Dennoch wies die in Kuala Lumpur erscheinende „New Straits Times" auf die Instabilität dieses im wesentlichen vom Ausland abhängigen Industriezweiges hin. Nur zehn Prozent der verarbeiteten Rohstoffe kommen aus Malaysia, und auf technologischem Gebiet verfügt das Land nicht über die notwendigen Kapazitäten und Kenntnisse. Außerdem, so wurde in der Zeitung gefordert, müsse die Elektronikindustrie von der Herstellung einzelner Komponenten zur Fertigung elektronischer Gesamtsysteme übergehen, wenn sich dieser Industriesektor fest im Land etablieren soll. Sonst bestehe die Gefahr, daß die niedrigen Löhne das einzig Interessante für ausländische Unternehmen blieben.

ADN

Dialog

Zum Beitrag „Sequentielle Online-Verarbeitung von dBASE-II-Dateien mit TURBO-PASCAL" in MP 6/1988, S. 179

Zu der gleichen Zielstellung schlage ich eine Methode vor, bei der die Hochsprache nicht verlassen und nicht auf ein Assemblerprogramm zurückgegriffen wird.
Im Bild 1 wird ein PASCAL-Quelltext gezeigt, der dem Test diente, in TURBO-PASCAL geschriebene Programme als Unterprogramme für dBASE-Anwendersoftware zu nutzen. Der Test wurde z. Z. noch nicht auf eine Parameterübergabe ausgedehnt.
Die vorgestellte Methode bietet die Möglichkeit, laufzeitintensive Programmabschnitte des dBASE-Programms durch schnellere Programme abzulösen. Nach Abarbeitung wird vom Unterprogramm (Start aus dBASE mit QUIT TO „COMFILE") wieder das Datenbanksystem nebst Anwenderprogramm gestartet. Wegen der z. Z. fehlenden Parameterübergabe können nur solche Programmabschnitte ersetzt werden, die unabhängig von globalen Parametern arbeiten. Das Anwenderprogramm wird von Neuem gestartet, hingegen bei realisierter Parameterübergabe eine als Unterprogramm fungierende Kommandodatei gestartet werden kann. Zur Übergabe von Parametern könnte der E/A-Puffer des Betriebssystems oder die von dBASE gebotene Möglichkeit des Parameterfiles genutzt werden.
Mit dieser Methode ist es möglich, dBASE-Software schrittweise, für den Anwender kaum spürbar – außer im Laufzeitverhalten –, zu ersetzen.

Detlef Knut, Berlin

```
program dbtest;
var db:file;
    cmd:string(.8.) absolute○80;
begin
  cmd:='MATERIAL';
  write('turbo-programm');
  assign(db,'DB2.COM');
  execute(db);
end.
```

***Bild 1** Aufruf des Datenbanksystems und Anwenderprogramms*

Zu den Beiträgen „Arbeit mit ASCII-Dateien" sowie „Arbeit mit Direktzugriffsdateien" in MP 5/88, S. 133 u. S. 154

Eine problemlose Verwendung von BASIC-DA-Dateien unter anderen Programmiersprachen ergibt sich durch Vergrößerung des Pufferbereiches um 2 Byte, die konstant mit 0DH 0AH gefüllt werden.
Hinter dem letzten Datensatz ist ein weiterer Satz auszugeben, nachdem das erste Byte des Puffers mit LSET X○ = CHR○(26) gefüllt wurde. Diese Datei kann in BASIC und anderen Programmiersprachen als sequentielle Datei einschließlich der EOF-Funktion verwendet werden.
Für DA-Dateien ist eine Speicherung der Datensatzanzahl erforderlich, da die EOF-Funktion nicht anwendbar ist. Diese Registrierung erfolgt zweckmäßig in einer Hilfsdatei, in der weiterhin alle notwendigen Merkmale für das gesamte BC-Projekt gespeichert werden können.
Zu Beginn des Programmlaufes wird diese Datei eröffnet, eingelesen und abgeschlossen. Die gelesenen Werte werden mit vorgegebenen Parametern verglichen. Bei Nichtübereinstimmung wird die Diskette abgelehnt und eine andere angefordert.
Beispiel für den Aufbau einer Hilfsdatei:

1. Satz Diskettennummer (6 Byte), Projektkennzeichen (6 Byte)
2. Satz Diskettenkennzeichen , Datum letzte Bearbeitung (ttmmjj)
3. Satz Satzanzahl Datei 1, Satzanzahl Datei 2
4. Satz Satzanzahl Datei 3, Satzanzahl Datei 4

.
.
.

aufgebaut als DA-Datei, je Datensatz 12 Byte Zeichenkette.
An markanten Programmpunkten, spätestens zum Programmende, ist die Datei zu aktualisieren. Durch geeigneten Aufbau dieser Datei ist es auch möglich, große Dateien, die sich über mehrere Disketten erstrecken, zu verarbeiten. Dazu sind ein Reihenfolgekennzeichen einzuführen und die Grenzen des Identbegriffes der Teildateien in der Hilfsdatei zu speichern. Die Hilfsdatei der ersten Teildatei muß alle Grenzen enthalten. Nach Eingabe des Suchbegriffes muß die Kontrolle der Grenzen und darauf die Kontrolle des Reihenfolgekennzeichens erfolgen. Bei positivem Ergebnis erfolgt die Verzweigung zum Suchlauf, bei negativem Ergebnis die Aufforderung zum Diskettenwechsel mit anschließender Kontrolle der Diskette.

Klaus Hildebrandt, Magdeburg

Maschinenbau und Elektronik zu neuer Qualität verbunden

Mit dem Entwurf anwenderspezifischer Schaltkreise für Erzeugnisse des Schwermaschinen- und Anlagenbaus hat der VEB Forschung, Entwicklung und Rationalisierung (FER) Magdeburg seit 1987 eine neue wissenschaftlich-technische Aufgabe übernommen. In diesem Jahr entstehen acht Schaltkreisentwürfe für Kombinate des Industriezweiges. Die Produktion der Bauelemente erfolgt dann in dafür spezialisierten DDR-Betrieben. „Die eigene Entwurfstätigkeit versetzt uns in die Lage, Maschinenbau und Elektronik in einer neuen Qualität zu verbinden, weil wir die Voraussetzungen für die Automatisierung der Erzeugnisse selbst schaffen", sagte FER-Direktor Dr. Klaus Hieckmann in einem ADN-Gespräch. Bis 1990 soll im Schwermaschinen- und Anlagenbau in 65 Prozent der Erzeugnisse Mikroelektronik integriert sein. Bei Neuentwicklungen sind 90 Prozent vorgesehen. Solche Steuer- und Prozeßüberwachungssysteme ermöglichen, die Masse der „schweren Brocken" im Maschinenbau zu reduzieren und sind mit ausschlaggebend für ihre Konkurrenzfähigkeit auf dem Weltmarkt. So kommen aus diesem Betrieb des Schwermaschinenbaukombinates „Ernst Thälmann" die Schaltkreisentwürfe für neue Steuerungen, die für den Einsatz in Kabel- und Verseilmaschinen bestimmt sind. Im Magdeburger SKET-Stammbetrieb wurde parallel dazu begonnen, Leiterplatten als Schaltkreisträger selbst herzustellen. Um den rasch wachsenden Bedarf des Industriezweiges an Schaltkreisen zu decken, erhält FER Magdeburg bis 1991 ein modern ausgestattetes Entwurfs-, Entwicklungs- und Beratungszentrum. In ihm können Betriebe des Schwermaschinenbaus und aus dem Territorium komplette Schaltungen erarbeiten lassen oder die Computertechnik dieses Zentrums für selbständige Entwurfsarbeit nutzen. Dr. Hieckmann erklärte, daß die Arbeit im FER vorlauforientierten Charakter habe. „Wir schaffen heute die Komponenten für den bedienarmen Betrieb der Zukunft", dazu zählen Automatisierungslösungen für produktionsvorbereitende und Produktionsprozesse. Rund 420 Mitarbeiter befassen sich mit der Mikroelektronik, Industrierobotertechnik, der computergestützten Konstruktion und Produktion, der Optoelektronik und Lasertechnik.

ADN

Bildungscomputer robotron A 5105

Dr. Gert Keller,
Dr. Gunter Kleinmichel,
VEB Robotron-Meßelektronik
„Otto Schön" Dresden

Im VEB Robotron-Meßelektronik „Otto Schön" Dresden wurde die Entwicklung des neuen Bildungscomputers robotron A 5105 abgeschlossen. Erste Muster dieses leistungsfähigen Gerätes werden derzeit dem Bildungswesen zum Test zur Verfügung gestellt.
Die Entwicklung des Bildungscomputers (BIC) wurde notwendig, da die bisher verfügbaren Kleincomputer der Serien KC 85 und KC 87 (siehe auch 4. Umschlagseite) die Ansprüche an eine niveauvolle und moderne Ausbildung nur teilweise erfüllen. Diese Computer wurden ursprünglich für andere Haupteinsatzgebiete konzipiert.
Mit dem BIC soll nun die Grundlage geschaffen werden für eine einheitliche Ausbildungsbasis im Fach Informatik, sowohl in den allgemeinbildenden Schulen, den EOS und den Spezialschulen als auch in den unterschiedlichen Bildungseinrichtungen der Berufsausbildung.

Pädagogisch-technische Forderungen an den BIC

Ausgehend von den zentralen Ausbildungszielen
– Herausbildung eines prinzipiellen Verständnisses für die Arbeitsweise eines Computers
– Vermittlung der Grundzüge des Programmierens
– Erlernen eines sicheren Umgangs mit Standard-Anwendersoftware
wurden die pädagogisch-technischen Forderungen an den Bildungscomputer zwischen dem Ministerium für Volksbildung, dem Staatssekretariat für Berufsbildung und dem VEB Robotron-Meßelektronik „Otto Schön" Dresden abgestimmt und folgende Schwerpunkte herausgearbeitet:

Konstruktion

– Die Konstruktion sollte robust, aber formschön sein und die gültigen ergonomischen Forderungen (Tastaturhöhe, Betrachtungsabstand) erfüllen.
– Aus Gründen der begrenzten Aufstellungsfläche wurde eine 31-cm-Bildröhre (schwarzweiß oder Farbe) gefordert.
– Einsatz einer modernen Computertastatur

Software

– Bereitstellung einer modernen, den internationalen Tendenzen entsprechenden BASIC-Version für die Grundausbildung und Unterstützung einer komfortablen Grafik durch dieses BASIC
– Möglichkeit der Abarbeitung der SCP-Standardsoftware des PC 1715, das heißt, Kompatibilität bezüglich der Systemschnittstellen und der Diskettenformate zu diesem Rechner

Hardware

– Unterstützung der für die Lauffähigkeit der geforderten Software erforderlichen ROM- und RAM-Speicherbereiche
– Unterstützung einer leistungsfähigen Bildausgabe für Text und Grafik entsprechend üblichen Standards (auch in Farbe), u. a. auch das wichtige Format 25 Zeilen × 80 Zeichen für SCP-Software
– Integration eines Diskettenlaufwerkes 5,25″
– Bereitstellung zahlreicher Interfaces zur Sicherung einer vielseitigen und flexiblen Ausbildung; Anschlußmöglichkeiten für Drucker, Plotter, Schülerexperimentiergeräte, zusätzliches Farbfernsehgerät, Kassettenmagnetbandgerät, Steuerhebel und diverse Module
– Verkopplungsfähigkeit für Computerkabinette

Details

– Umlaute im Standardzeichensatz und auf der Tastatur
– Programmierbarer Zeichensatz, auch für Quasigrafik
– Programmierbare Funktionstasten
– Übernahme bewährter Anweisungen des HC-BASIC des KC 85/1 in das BASIC des BIC, z. B. WINDOW und PAUSE
– Anschlußmöglichkeit für Fernsehgeräte, die in den Schulen bereits vorhanden sind.
Entsprechend der Festlegung für die Preisobergrenze wurde vereinbart, den Bildungscomputer auf der Basis der 8-Bit-Schaltkreisfamilie des U 880 aufzubauen.

Kurzbeschreibung des Entwicklungsergebnisses Bildungscomputer

Hardware und Konstruktion

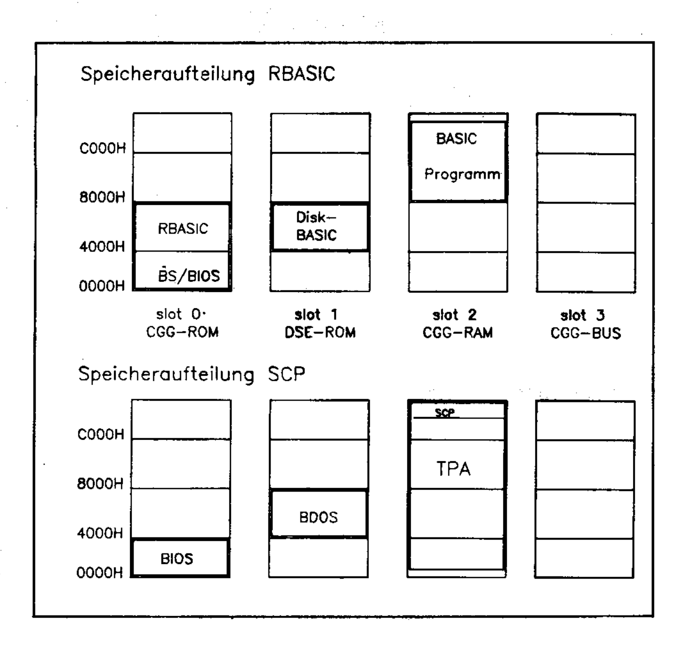
Der BIC besteht aus den 3 Gerätekomponenten

Computergrundgerät	(CGG)
Diskettenspeichereinheit	(DSE)
Monitor	(MON),

die mechanisch fest miteinander verbunden sind. Im **CGG** ist unter einer standardgerechten Computerflachtastatur eine Leiterplatte mit den Baugruppen Prozessor, Arbeits-, Bild- und Programmspeicher, BUS- und Speicherverwaltung, Bildausgabe und Kassetteninterface untergebracht. Neben einer BUS-Schnittstelle in einem speziellen Modulschacht und den Anschlüssen für 2 Steuerhebel und ein Magnetbandgerät sind zwei Steckverbinder zur Kopplung des CGG an die DSE vorhanden.
Die **DSE** enthält die Stromversorgungsbaugruppe für den BIC, ein Diskettenlaufwerk 1.6 und auf einer großen Leiterplatte die Ansteuerbaugruppe für das Diskettenlaufwerk und zahlreiche Interface-Ansteuerungen, z. B. serielle und parallele Datenausgänge für Anschluß von Drucker, Plotter und Schülerexperimentiergerät, Lokalnetzanschluß und Anschlüsse für unterschiedliche Bildausgabegeräte.
Der **Monitor** enthält eine monochrome 12-Zoll-Bildröhre und ist auf der DSE dreh- und schwenkbar befestigt. Basis der Prozessoreinheit ist der 8-Bit-CPU-Schaltkreis U 880 der 4-MHz-Baureihe. Der interne Gerätebus ist, wie bei den robotron-Kleincomputern, elektrisch und konstruktiv weitgehend zum K-1520-Bus kompatibel und sowohl zum Modulschacht des CGG als auch an einen BUS-Steckverbinder (an der Rückseite der DSE) herausgeführt. Der BUS ist vollständig getrieben und kurzschlußfest.

Wichtige technische Daten

Prozessor:	**UA 880 D**
Videocontroller:	U 82720
Floppy-Disk-Controller:	**U 8272**
Taktfrequenz:	3.75 MHz
Speicherkapazität:	48 (oder 64) KByte ROM mit BASIC-Betriebssystem, Floppy-Treiber und Systemprogrammen 64 KByte RAM Anwender-Arbeitsspeicher 64 K × 16 Bit RAM [illegible] speic[illegible]
Massenspeicher:	Diskette 5 1/4 Zoll Kassettenmagnetband (mit Motorschaltspannung)
Tastatur:	Computerflachtastatur (8-Bit-Zeichensatz)
Bildausgabe:	Monitor 12 Zoll, monochrom oder Fernsehgerät s/w über HF-Modulator oder Farbmonitor und Farbfernsehgerät über RGB-Ausgang
Bildaufbau	*alphan[illegible]che Ausgabe mit* [illegible] Zeilen [illegible]0 Zeichen 25 Zeilen [illegible]0 Zeichen 16 Vorderg[illegible]nd-, 8 Hintergrundfarben *Grafikausgabe* [illegible] 320 × 200 Bildpunkten in 4 aus 16 Farben 640 × 200 Bildpunkten in 4 aus 16 Farben 320 × 200 Bildpunkten mit 16 Farben je Punkt
Tonausgabe:	3 Kanäle, 8 Oktaven über Audioausgang
Weitere externe Anschlüsse:	– Modulsteckplatz – BUS-Anschluß – V.24-Druckerinterface (unidirektional) – V.24-Plotterinterface (bidirektional) – 2 × 8-Bit-Parallel-Interface (zum Anschluß Schülerexperimentiergerät und als CENTRONICS-Interface programmierbar) – Netzanschluß zur Einkopplung in Computernetze (ROLANET-kompatibel) – 2 Steuerhebelanschlüsse – Anschluß [illegible] Diskettenbeistellgerät (maximal 2 Laufwerke)
Software:	**RBASIC** im ROM (48 KByte – **SCREEN**-Editor – 14 Stellen Rechengenauigkeit [illegible] omfortable Grafik[illegible]ungen [illegible]equentielle und Direktzugriffsdateien auf Diskette **SCPX 5105** als **RAM**-Betriebssystem – [illegible]kompatibel zum **SCP** [illegible] von **PC 1715** [illegible]kette mit wich[illegible] [illegible]rogrammen [illegible]etten- und Datei[illegible] [illegible] für beide Syste[illegible]
Maße: (B × T × [illegible]	[illegible]grundger[illegible] [illegible] 220 mm [illegible] [illegible]eicher[illegible]einheit [illegible]mm [illegible] 135 m[illegible]
M[illegible]	



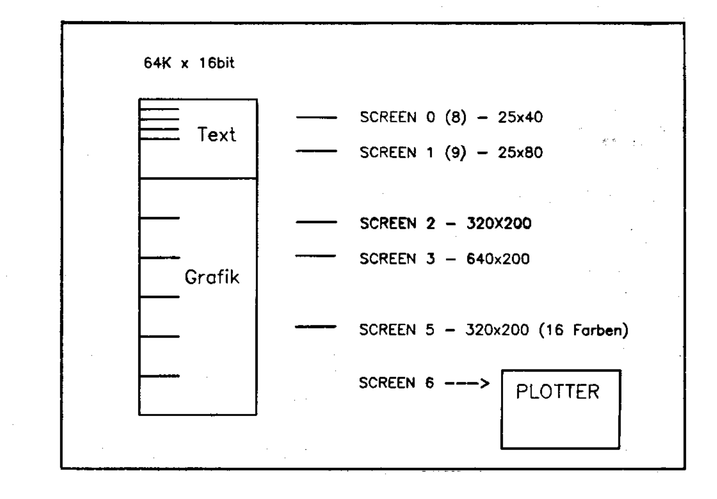


Bild 2 Bildspeicher und SCREENs

◀ ***Bild 1 Speicheraufteilung beim A 5105 für RBASIC und SCP***

Die Speicherkonfiguration besteht aus 64 KByte Arbeitsspeicher, 64 K × 16 Bit Bildspeicher und mindestens 48 KByte Programmspeicher. Der adressierbare Speicherraum wird mittels 4facher SLOT-Umschaltung auf 256 KByte erweitert. Die Anordnung innerhalb dieser 4 Bereiche zu je 64 KByte zeigt Bild 1. Die Speicherverwaltung sowie ein Audiogenerator, die Tastaturansteuerung und die Adreßdekodierung für die E/A-Ports sind in einem kundenspezifischen Schaltkreis (ASIC) realisiert.
Für die Bildausgabeeinheit werden der Grafik-Display-Controller U 82720 und ein weiterer kundenspezifischer Schaltkreis eingesetzt. Damit wird eine hochleistungsfähige Grafikdarstellung mit bis zu 16 Farben je Pixel ermöglicht. Diese Farbdarstellung ist natürlich nur bei alternativem oder zusätzlichem Anschluß eines Farbmonitors bzw. eines Farbfernsehgerätes mit RGB-Signaleingang nutzbar. Die erforderlichen Signalausgänge sind an der DSE vorhanden.
Mit dem Floppy-Disk-Controller U 8272 wird die Ansteuerbaugruppe für maximal 3 Laufwerke 1.6 aufgebaut, wobei an der DSE der Anschluß für 2 weitere externe Laufwerke vorhanden ist. Die Stromversorgungseinheit ist als Schaltnetzteil aufgebaut und liefert neben den benötigten Versorgungsspannungen von +5 V, +12 V und −12 V bei +5 V eine Stromreserve für den Nutzer von zirka 1 A.

Software

Entsprechend den sehr unterschiedlichen Einsatzforderungen werden für den Bildungscomputer zwei Betriebssysteme bereitgestellt, die beide mit den im SCP üblichen Disketten- und Dateiformaten arbeiten.

Betriebssystem RBASIC

Das System RBASIC ist in 48 KByte ROM des Computers enthalten und stellt neben zahlreichen auch im SCP nutzbaren Treiberprogrammen (für Tastatur, Bildausgabe, Interfaces) einen leistungsfähigen BASIC-Interpreter bereit.
Im RBASIC stehen etwa 150 BASIC-Kommandos, Anweisungen und Funktionen zur Verfügung, die in Syntax und Wirkungsweise mit denen des BASI des Personalcomputers robotron EC 1834 kompatibel sind.

Einige Leistungsmerkmale des RBASIC:
- SCREEN-Editor
- WINDOW-Anweisung für Text und Grafik
- 3 Zahlenformate (maximal 14 Stellen Genauigkeit)
- Unterstützung verschiedener Bildformate

Text:
25 × 40 und 25 × 80 Zeichen (je Zeichen 16 Farben)

Grafik:
320 × 200 und 640 × 200 Punkte (je Punkt 4 aus 16 Farben)
320 × 200 Punkte (je Punkt 16 Farben)
Die Aufteilung des Bildspeichers ist Bild 2 zu entnehmen.
- Arbeit mit bis zu 16 Text- und 6 Grafikbildern
- Arbeit mit sequentiellen Dateien (Diskette und Kassette)
- Arbeit mit Zugriffsdateien (Diskette)
- Kommandos zur Dateiverwaltung auf Diskette
- Hardcopy-Programm für Text und Grafik integriert.

Betriebssystem SCPX 5105

Das Betriebssystem SCPX 5105 arbeitet im 64-KByte-RAM-Bereich des Computers. Es ist vollständig kompatibel zum SCPX 1715 des in der DDR weitverbreiteten Personalcomputers robotron 1715, so daß alle Anwenderprogramme, die die Betriebssystemschnittstellen des robotron 1715 einhalten, auch auf dem Bildungscomputer robotron A 5105 lauffähig sind.

Spezielle Merkmale des SCPX 5105:
- TPA größer als 48 KByte
- Vorzugsweise Unterstützung des Bildformates 25 × 80 Zeichen
- Spezielles Dienstprogramm zum rationellen Kopieren von Dateien mit nur einem Diskettenlaufwerk
- Möglichkeit der Nutzung von BIOS-Rufen des RBASIC (z. B. für Grafikroutinen im PASCAL).

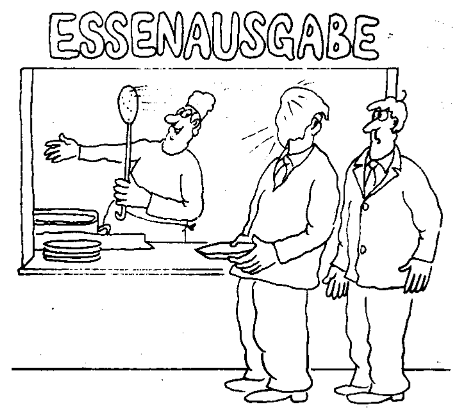
Besondere Merkmale der Entwicklung

Die Zielstellung der Auftragsentwicklung des Bildungswesens war die Erreichung eindeutig geforderter hoher Leistungsparameter unter Einhaltung vorgegebener ökonomischer Bedingungen. Dabei konnte die Erfüllung des Forderungsprogrammes nur durch Verwendung der modernsten verfügbaren Bauelemente und unter Einsatz rationellster Fertigungstechnologien gesichert werden, auch wenn dadurch häufig technologisches Neuland betreten werden mußte und überdurchschnittlich hohe Aufwendungen in der Fertigungsvorbereitung erforderlich waren. Wegen des zeitlichen Gleichlaufes von Geräteentwicklung und Entwicklung kundenspezifischer Schaltkreise waren die Arbeiten mit einem hohen Risiko verbunden.
Mit dem Einsatz der Bestückungstechnologie für aufsetzbare Bauelemente, dem digitalen In-Circuit-Test als wesentlichstem Prüfverfahren und der Verwendung von Vollplastgehäusen ist die Basis für eine rationelle Fertigung in hohen Stückzahlen gegeben.
Bei der Softwaregestaltung war schließlich immer die Grundforderung des Bildungswesens zu berücksichtigen, daß der BIC in Softwaregestaltung und Bedienphilosophie einen hohen Grad an Kompatibilität zu den Computern aufweisen soll, die der Auszubildende bei seinem praktischen Berufseinsatz vorfinden wird.

✉ KONTAKT ☎

VEB Robotron-Meßelektronik „Otto Schön" Dresden, Lingnerallee 3, PSF 211, Dresden, 8010; Tel. 4875846 (Dr. Keller)

Ausgabegeschwindigkeit



Zeichnung: Frank Steger

dBASE III im Vergleich

Dr. Wilfried Grafik, Bärbel Osten
Humboldt-Universität zu Berlin, Sektion Mathematik

Ein wesentliches Merkmal von Datenbanksprachen ist die Komplexität ihrer Kommandos, dBASE enthält viele komplexe Kommandos, die in höheren Programmiersprachen wie PASCAL, C oder MODULA nur durch mehr oder weniger komplexe Prozeduren ausgedrückt werden können. Hinzu kommt, daß Datenbankkommandos oft Fileoperationen beinhalten, also den Zugriff zu einem externen Speicher. Die Geschwindigkeit, mit der Kommandos abgearbeitet werden, wird also erheblich durch die Effektivität der Filearbeit bestimmt. Diese wiederum ist bedingt durch die Geräte und Datenträger (Disketten oder Harddisk), ihre Controller, ihre Treiber und durch Anzahl und Größe der verfügbaren Puffer.

Fragestellung

Welche Leistungen bringt dBASE III in Abhängigkeit von den benutzten Geräten? Welchen Einfluß hat dabei die Prozessorgeschwindigkeit? Wie ist das Verhältnis zu vergleichbaren Kommandos in dBASE II? Diese Fragen wollen wir im folgenden durch Messungen beantworten.
Betrachten wir zuvor noch andere interessante Fragestellungen, z. B. nach dem Einfluß der Länge der Datensätze auf die Ausführungszeiten. Untersuchungen an drei verschiedenen Datenbanken mit Sätzen unterschiedlicher Länge lieferten hier Unterschiede, die aber keine Proportionalität der Ausführungszeiten zur Länge der Datensätze erkennen ließen. Umgekehrt kann also nicht geschlußfolgert werden, daß die Einsparung einiger Bytes in einem Datensatz eine wesentlich schnellere Ausführung der Kommandos bewirkt. Hier sollte man unbedingt problemorientiert vorgehen, das heißt, die Struktur der Datensätze wird durch die zu lösende Aufgabe bestimmt, nicht durch die Eigenschaften irgendwelcher gerade verfügbarer Systeme.
Ferner wurde die Sortierzeit eines Datenbankfiles in Abhängigkeit von seiner Länge untersucht. Dabei stellte sich heraus, daß es in dem gemessenen Bereich von 500 bis 28 000 Datensätzen mit je 56 Byte keine sprunghaften Veränderungen der Sortierzeit gibt. Das Datenbankfile mit 500 Sätzen belegt 29 KByte. 28 000 Datensätze belegen 1568 KByte. Die Sortierzeit beträgt 0:0:20 bzw. 1:05:53 Stunden, was einen Faktor von etwa 198 ergibt. Diese Messungen wurden auf einem IBM-PC/XT durchgeführt. Wir wollen diese Abhängigkeit hier nicht näher untersuchen.
Einige Einflüsse auf die Messungen können nicht exakt ausgewiesen werden bzw. müssen unberücksichtigt bleiben. So ist die Frage zu stellen, welchen Einfluß die Länge der Operanden, die in den dBASE-Kommandos verarbeitet werden, haben. Sind viele Operationen ohne Filezugriffe auszuführen, so sind schnelle Prozessoren im Vorteil.
Welchen Einfluß hat die Größe des verfügbaren Hauptspeichers? Besteht ein signifikanter Unterschied bei 512 oder 640 KByte Speicherplatz? Stört oder nutzt eine RAM-Floppy? Versuche mit einem XT (640 KByte und 4,75 MHz) und einem Schneider 1512 (512 KByte und 8 MHz) weisen aus, daß ein in diesen Größen variierender Speicher kein Nachteil ist. Eine RAM-Floppy kann für dBASE nur dann nützlich sein, wenn sie Kommando- oder Formatfiles enthält. Für das Sortieren und Indizieren ist sie praktisch nicht einsetzbar.
Wir sind uns darüber im klaren, daß die Messungen durch andere, hier nicht bewußt gesteuerte Einflüsse, geringfügig verzerrt werden. Ein derartiger Einfluß ist z. B. die Verteilung der Daten auf der Platte. Ist die Datei kontinuierlich abgespeichert? Wie groß ist ihre Entfernung vom Fileverzeichnis, wieviel Köpfe hat die Platte usw.? Diese Randbedingungen beeinflussen offenbar die Positionierzeit der Köpfe. Es ist aber gerechtfertigt, diese Einflüsse nicht zu betrachten, da man unter Produktionsbedingungen auch keinen Einfluß darauf nehmen wird. Das Ziel dieses Beitrags besteht also nicht darin, generelle Benchmark-Tests für dBASE zu entwickeln und durchzuführen.
Bekannt sind Messungen zu REDABAS, wie sie z. B. in /1/ veröffentlicht sind. Wir wählen für unsere Untersuchungen ähnliche, charakteristische und komplexe Kommandos von dBASE III und testen sie separat. Sie sind charakteristisch im Sinne ihrer Existenz im Datenbanksystem dBASE, nicht bezüglich ihrer Anwendungshäufigkeit in praxisrelevanten Programmen. Das Ziel besteht darin, die Abhängigkeit der Geschwindigkeit von dBASE von den verwendeten Geräten auszuweisen.
Alle Messungen werden an einem Datenbankfile durchgeführt, das im praktischen Einsatz ist. Es dient der Verwaltung von Zeitschriftenartikeln in unserer Sektion. Die Länge des Datenbankfiles wird auf 1000 Datensätze normiert, was den Vergleich mit anderen Messungen erleichtert. Für diesen Test wählen wir ein Datenbankfile **journ.dbf**, dessen Struktur in Tafel 1 wiedergegeben ist.
Dieses Datenbankfile benötigt auf der Diskette für 1000 Datensätze 214 KByte. Es ist auf einer 720-KByte-Diskette gerade noch sortierbar, da beim Sortieren zwei Hilfsfiles angelegt werden, von denen zumindest eins die Größe des Datenbankfiles erreicht.
Uns interessiert zuerst die Ausführungszeit in Abhängigkeit vom benutzten Speichermedium. Welchen Einfluß hat die Benutzung von Disketten im Vergleich zum Einsatz von Festplatten? Unterscheiden sich die Zeiten bei der Benutzung von 40-Spur- und 80-Spur-Laufwerken? Dazu führen wir die gleichen Messungen mit den verschiedenen Datenträgern durch. Die benutzten Disketten enthalten keine weiteren Daten, so daß die Kopfbewegungen minimal sind.

Neun verschiedene Rechner

Inwieweit hängt die Geschwindigkeit von dBASE vom verwendeten Rechnertyp ab? Wir benutzen einen **IBM-PC/XT-kompatiblen** Rechner, einen **AT-kompatiblen** Rechner, einen **AT**, einen **„Schneider 1512"**, einen **A 7100**, einen **A 7150**, einen **EC 1834** und einen durch eine Subprozessorkarte erweiterten **8-Bit-OEM-Rechner.** Zum Vergleich mit der Leistungsfähigkeit von dBASE II werden die gleichen Kommandos auf einem **A 5120**, einem A 7100 und auf dem erweiterten OEM-Rechner abgearbeitet.
Der XT enthält einen Prozessor I 8088, der mit 4,75 MHz betrieben wird, 640 KByte Speicher, Diskettenlaufwerke mit 40 und 80 Spuren und eine Festplatte von 20 MByte.
Der AT-kompatible Rechner enthält einen Prozessor I 80286, der mit 10 MHz betrieben wird, 640 KByte Speicher, 1,2-MByte-Diskettenlaufwerke und eine Festplatte von 20 MByte.
Der AT 0.2 enthält einen I-80286-Prozessor (6 MHz), 3,2 MByte Speicher und eine Festplatte von 110 MByte. Unter einem UNIX-Betriebssystem werden die Messungen mit einer dBASE-III-Weiterentwicklung (FoxBase+) durchgeführt.
Der Schneider 1512 enthält einen I-8086-Prozessor (8 MHz), 512 KByte Speicher und eine 32-MByte-Festplatte.
Der A 7150 und der EC 1834 enthalten einen Prozessor K 1810 WM86, der mit 4,9 MHz betrieben wird, 512 KByte bzw. 640 KByte Speicher und eine 50-MByte-Festplatte.
Der zu einem 16-Bit-Rechner erweiterte OEM-Rechner besitzt eine Subprozessorkarte mit einem I 8086, der mit 8 MHz betrieben wird, und 512 KByte Speicher. Die Ein- und Ausgabe wird über den 8-Bit-Teil des A 5120 realisiert. 8- und 16-Bit-Teil sind über den K-1520-Bus miteinander verbunden. Als Laufwerke werden 80-Spur-Laufwerke verwendet. Die zugehörigen Gerätetreiber sind speziell auf diese Geräte zugeschnitten, so daß eine maximale Geschwindigkeit im Diskettenzugriff erreicht wird. MS-DOS und CP/M 86 sind auf diesem Rechner lauffähig. Der Test unter MS-DOS mit dBASE III gestattet einen Vergleich zu anderen MS-DOS-Rechnern. Der Test unter CP/M 86 mit dBASE II gestattet einen Vergleich zum Bürocomputer und zum A 7100. Da in dBASE II das Sortieren nur nach einem Feld möglich ist, fehlen die Werte in der Zeile für das Sortieren nach drei Feldern.
Der verfügbare A 7100 ist mit dem Prozessor K 1810 WM86 ausgerüstet und besitzt einen Speicher von 512 KByte und Diskettenlauf-

Tafel 1 Datenbankstruktur – LIST STRUCTURE des Files journ.dbf

Anzahl der Datensätze – 1000
Letztes Änderungsdatum – 12.11.87

Feld	Feldname	Typ	Länge	Dez
1	KURZTITEL	Zeichen	40	
2	TITEL	Zeichen	90	
3	ORT_VERLAG	Zeichen	30	
4	JAHRG	Zeichen	8	
5	STANDORTE	Zeichen	30	
6	H1	Numerisch	3	
7	H2	Numerisch	3	
8	H3	Numerisch	3	
9	H4	Numerisch	4	
** Gesamt **			212	

Tafel 2 Ausführungszeiten für XT und AT

Zeiten in mm:ss mit verschiedenen Datenträgern

Kommando	XT auf Festplatte	XT auf Floppy 40 Spuren	XT auf Floppy 80 Spuren	XT auf Festplatte; Clipper	AT-kompat. auf Festplatte	AT-kompat. auf Floppy	AT auf Festplatte; UNIX; FoxBase+
COUNT	0:29	1:33	1:33	0:12	0:20	1:14	0:25
DELETE FOR	0:20	1:51	1:44	0:13	0:12	1:14	0:07
INDEX 1	5:15	—[1]	18:35	2:53	3:23	7:18	0:58
INDEX 3	5:22	15:18	19:51	2:41	3:43	12:55	0:41
LOCATE FOR	0:18	1:32	1:32	0:12	0:09	1:11	0:06
PACK	1:00	3:01	5:38	0:09	0:45	2:24	0:30
REPLACE ALL	1:13	3:07	5:42	1:16	0:45	2:20	0:35
SORT 1	5:35	—[1]	20:29	5:17	4:00	11:45	1:03
SORT 3	6:59	—[1]	22:54	7:17	4:13	10:38	0:46
SUM 1	0:40	1:33	1:32	0:15	0:21	1:15	0:22
SUM 5	1:03	1:33	1:33	0:42	0:23	1:14	0:34

[1] Nicht genügend Platz auf einer Diskette mit 360 KByte

Tafel 3 Ausführungszeiten für "Schneider", OEM-Erweiteung, A 7150, EC 1834 und Clipper unter MS-DOS

Zeiten in mm:ss

Kommando	Schneider 1512 auf Festplatte	A 5120+8086 Floppy, MS-DOS	A 7100 Floppy, MS-DOS, III+	A 7150 auf Festplatte	EC 1834 auf Festplatte
COUNT	0:23	1:23	1:36	0:32	0:17
DELETE FOR	0:16	1:30	1:48	0:20	0:17
INDEX 1	3:07	11:06	5:43	5:15	3:34
INDEX 3	3:21	12:49	5:07	5:56	3:40
LOCATE FOR	0:15	1:23	1:35	0:18	0:15
PACK	0:40	2:56	4:54	1:05	0:37
REPLACE ALL	0:41	2:45	5:08	1:17	0:45
SORT 1	3:21	14:25	13:01	5:36	4:02
SORT 3	4:12	14:36	14:31	6:48	4:59
SUM 1	0:26	1:26	1:40	0:41	0:23
SUM 5	0:39	1:25	3:03	1:03	0:43

werke für 80 Spuren. Auf ihm wurde ein MS-DOS-Betriebssystem installiert, unter dem dBASE III+ läuft. Ein weiterer Test auf diesem Rechner erfolgt mit dem Betriebssystem SCP 1700 und dBASE II.

Der 8-Bit-Bürocomputer A5120 hat einen Speicher von 64 KByte, einen Prozessor U880, der mit 2,5 MHz betrieben wird, und ebenfalls Diskettenlaufwerke mit 80 Spuren. Er läuft mit einem CP/M-Betriebssystem der Humboldt-Universität (s. auch MP-Börse).

Tests

Zum Test werden auf allen Rechnern die folgenden Kommandos in der angegebenen Reihenfolge abgearbeitet:

COUNT
SUM h1
SUM h1, h2, h3, h4, h1+h2+h3+h4
LOCATE FOR RECNO()=1000
REPLACE ALL titel WITH titel
DELETE ALL FOR titel="xxx"
PACK
SORT ON titel TO x
SORT ON titel, ort_verlag, standorte TO x
INDEX ON titel TO jotitel
INDEX ON ort_verlag + jahrg + standorte TO joausdr

Zu beachten ist die Länge des Feldes **titel** (90 Zeichen). Die maximale Länge eines Indexausdrucks beträgt 100 Zeichen. Deshalb werden im zweiten INDEX-Kommando andere Felder benutzt. Bei der Wahl der Kommandos kommt es nicht auf logisch zusammenhängende und sinnvolle Operationen an. So ist das getestete REPLACE-Kommando logisch nicht sinnvoll, gibt aber das Prinzip wieder: Ein Datensatz wird gelesen, teilweise ersetzt und zurückgeschrieben.

Tafel 2 vergleicht die Werte für den XT und den AT. Es war zu erwarten, daß die Arbeit mit der Festplatte wesentlich schneller ist, als die Verwendung von Disketten. Bemerkenswert ist dagegen, daß die Ausführungszeiten für die 40-Spur-Laufwerke zum Teil erheblich unter denen für 80-Spur-Laufwerke liegen. Ein AT kann die Geschwindigkeit seines Prozessors nicht ausspielen, wenn er mit Disketten arbeiten muß. In diesem Fall ist ihm selbst ein XT mit Festplatte überlegen.

Tafel 3 zeigt, daß die Meßwerte vom A7150 und vom XT fast gleich sind. Der Schneider wird durch seinen kleineren Hauptspeicher, verglichen mit dem XT, nicht merklich behindert. Er bringt deutlich bessere Zeiten, die zwischen den Werten des XT und des AT liegen. Der EC1834 ist bei diesen Messungen durchaus mit dem Schneider vergleichbar.

Die in Tafel 2 in der Spalte Clipper angegebenen Werte wurden ermittelt, indem das Kommandofile mit dem Clipper-Compiler übersetzt und anschließend gelinkt wurde. Dazu waren geringfügige Änderungen am Kommandofile erforderlich, die die Meßwerte jedoch nicht beeinflussen. Die Übersetzungszeit lag etwa bei 20 Sekunden. Der Linklauf dauerte mit **plink86** dagegen 4 Minuten, mit dem Linker des MS-C-Systems 1:15 Minuten. Es wurde ein .exe-File von 133872 Byte erzeugt. Das erscheint auf den ersten Blick viel. Bedenkt man jedoch, daß damit alle benutzten Funktionen realisiert werden und daß die Files **dBASE.exe** und **dBASE.ovl** 120257 bzw. 160768 Byte lang sind, so wird dieser Wert relativiert. Die Ausführungszeiten für das übersetzte Programm sind mit denen des AT vergleichbar. Lediglich beim Sortieren schneidet das vom Clipper erzeugte Programm schlechter ab. Das läßt andererseits darauf schließen, daß die in dBASE III implementierten Sortieralgorithmen leistungsfähig sind. In dBASE II wurden sie allgemein bemängelt, was sich auch in den Zeiten in Tafel 4 widerspiegelt.

In Tafel 4 sind Werte für dBASE II zusammengestellt. Erstaunlich ist hier die Geschwindigkeit des erweiterten OEM-Rechners. Er erreicht unter dBASE II auf Diskette fast die gleichen Werte, die der XT mit dBASE III auf der Festplatte liefert! Für die Versuche auf dem XT-Rechner ergeben sich folgende Größen für die Indexfiles und das Datenbankfile selbst:

Filename		Länge in Byte
JOURN	DBF	213504
JOTITEL	NDX	193024
JOAUSDR	NDX	140288

Erweitern wir die Datensätze unseres Datenbankfiles derart durch fiktive Felder, daß eine effektive Datensatzlänge von 1000 Byte zustande kommt und führen wir die gleichen Tests unter den ansonsten gleichen Bedingungen durch, so ergeben sich die in Tafel 5 zusammengestellten Werte.

Auf Diskette ist dieses Datenbankfile nur auf 1,2-MByte-Disketten speicherbar, also auch dort nicht mehr sortierbar oder indizierbar. Die erzeugten Indexfiles haben die gleiche Länge wie im vorangegangenen Test. Das war zu erwarten, da sich der Indexausdruck nicht verändert. Das Datenbankfile selbst nimmt immerhin über ein Megabyte ein!

Tafel 4 Ausführungszeiten für dBASE II

Zeiten in mm:ss unter CP/M 80 und 86 mit dBASE II

Kommando	A 5120+8086 auf Floppy; CP/M 86	A 7100 auf Floppy; SCP 1700	A 5120 auf Floppy; CP/M 80
COUNT	0:24	0:45	1:15
DELETE FOR	0:26	1:22	1:27
INDEX 1	5:03	1:50:55	1:12:49
INDEX 3	5:16	1:54:50	1:13:32
LOCATE FOR	0:25	1:23	1:12
PACK	0:52	9:57	8:36
REPLACE ALL	0:54	9:56	8:19
SORT 1	11:43	1:55:07	1:12:35
SORT 3	—	—	—
SUM 1	0:24	0:59	1:16
SUM 5	0:32	1:03	1:51

Tafel 5 Ausführungszeiten für ausgewählte Kommandos bei 1000 Datensätzen mit einer Satzlänge von 1001 Byte

Zeiten in mm:ss

Kommando	XT auf Festplatte
COUNT	1:08
DELETE FOR	1:02
INDEX 1	7:05
INDEX 3	7:50
LOCATE FOR	0:58
PACK	4:47
REPLACE ALL	4:56
SORT 1	26:19
SORT 3	26:29
SUM 1	1:16
SUM 5	1:42

Tafel 6 Zeiten in mm:ss auf dem P8000, Betriebssystem WEGA, Dantenbanksystem 'Ing_db', 1000 Datensätze (Tupel)

entspr. Kommando aus dBASE	P 8000 auf Festplatte
COUNT	0:32
DELETE FOR	0:20
INDEX 1	8:50
INDEX 3	2:28
LOCATE FOR	—
PACK	—
REPLACE ALL	1:43
SORT 1	2:20
SORT 3	2:39
SUM 1	0:24
SUM 5	0:36

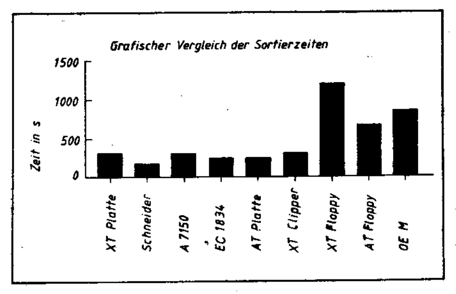


Bild 1 Grafischer Überblick über die Sortierzeiten nach einem Feld und für dBASE III

Ziehen wir eine Querverbindung zu den P-8000-Rechnern. Sie enthalten eine U-8000-CPU, die mit den anderen betrachteten Prozessoren nur insofern vergleichbar ist, als daß sie ebenfalls ein 16-Bit-Prozessor ist. Die unter MS-DOS bekannten Programme, also auch dBASE, laufen auf diesem Rechner nicht. Der benutzte P 8000 enthält 1 MByte Speicher und eine Festplatte von 44 MByte. Es sind drei Terminals angeschlossen. Als Betriebssystem läuft das UNIX-ähnliche WEGA. Auf dieser Konfiguration ist das Datenbanksystem **Ing_db** lauffähig. Die Kommandos von **Ing_db** liegen auf einer niedrigeren Ebene als die von dBASE. Es wurde versucht, die unter dBASE getesteten Kommandos abzubilden. Die konkreten Kommandos sollen im Detail nicht interessieren. Ansonsten wurden die gleichen Daten benutzt. Parallel zu diesen Tests liefen auf dem P 8000 keine CPU-intensiven Programme. Einige Messungen wurden mehrfach durchgeführt, um den Einfluß der parallel arbeitenden Nutzer abschätzen zu können. Dabei wurde bemerkt, daß Arbeiten mit dem Texteditor und kurze Übersetzungen die Werte nur unwesentlich beeinflussen. Für die Kommandos LOCATE und PACK aus **dBASE** konnten keine vergleichbaren Kommandos in **Ing_db** gefunden werden. Auch bei diesen Messungen wollen wir uns auf die Größenordnung der Werte konzentrieren (Tafel 6).

Welche generellen Schlußfolgerungen ergeben sich aus diesen Messungen?

① dBASE III enthält gegenüber dBASE II wesentlich schnellere Algorithmen zum Sortieren und zum Indizieren.

② Nicht nur die Prozessorgeschwindigkeit ist wesentlich. Die in einem Rechner eingebauten Geräte (Laufwerke, Festplatten), aber auch die Gerätetreiber sind wichtig. Zum 16-Bit-Prozessor gehört eine Festplatte, gehören leistungsfähige Gerätetreiber, mindestens 512 KByte Hauptspeicher und gute Algorithmen!

③ Berücksichtigt man, daß der A5120 nur einen 8-Bit-Prozessor enthält (2,5 MHz), daß er nur 64-KByte-Speicher hat und mit dem langsamen dBASE II arbeitet, so schließt er in diesem Vergleich nicht schlecht ab. Er demonstriert das Leistungsvermögen der 8-Bit-Klasse.

④ Die Hauptspeichergröße (640 K, 512 K oder künstlich verkleinert durch eine RAM-Floppy) hat keinen entscheidenden Einfluß.

⑤ Mit einem Compiler übersetzte dBASE-Programme erreichen auf dem XT eine Geschwindigkeit, die sogar dem AT mit dBASE in einigen Fällen überlegen ist. Es werden Optimierungen durchgeführt, die bei einer interpretativen Kommandoabarbeitung nicht möglich sind (vergleiche DELETE PACK). Die Vorzüge des Clipper-Compilers werden durch diese Messungen natürlich nicht erschöpfend ausgewiesen.

Mirecki /2/ untersucht drei dBASE-III-Compiler: **FoxBase, Clipper** /3/ und **Quicksilver.** Er faßt sein Ergebnis wie folgt zusammen: Die Geschwindigkeit eines Anwenderprogramms für dBASE III kann, im Durchschnitt betrachtet, durch Compilation nicht über den Faktor 2 gesteigert werden. Das gilt „unabhängig davon, wie schnell der compilierte Code ist".

⑥ Die durchgeführten Messungen beziehen sich im wesentlichen auf die Filearbeit. Dadurch werden die Vorteile schneller Prozessoren nicht sichtbar. Tests mit rechenintensiven dBASE-Programmen auf einem AT und mit übersetzten dBASE-Programmen stehen noch aus.

⑦ Die Übernahme von Datenbasen (Datenbankfiles) auf den P 8000 ist möglich. Die Umsetzung der Kommandos ist schwierig. Ein Transformator *dBASE nach Ing_db* ist noch nicht verfügbar. Generell fraglich ist die Überführung von Formatfiles, Berichten, Label-Files usw. Sind Daten und Kommandos übertragen, so weisen die durchgeführten Tests schnellere Zeiten auf als die der anderen getesteten Rechner.

⑧ Es existieren UNIX-kompatible Betriebssysteme, unter denen nach und nach Teile der weitverbreiteten MS-DOS-Software verfügbar gemacht wird, z. B. FoxBase+ als Weiterentwicklung von dBASE. Für dBASE-Nutzer ist es kein Problem, in solche UNIX-Umgebungen umzusteigen. Die Ausführungszeiten können bei geeigneter Hardware sogar wesentlich schneller sein als auf einem XT.

Literatur

/1/ Weber, P.: REDABAS-Laufzeiten unter Betriebssystemen für BC A 51xx und PC 1715. Rechentechnik/Datenverarbeitung 24 (1987) 4, S. 31

/2/ Mirecki, T.: Dialects of dBASE. PC TECH Journal, April 1987

/3/ Clipper Compiler – Anwender-Handbuch. Culver City: Nantucket Corporation 1985

✉ KONTAKT ☎

Humboldt-Universität, Sektion Mathematik, Bereich IV, PSF 1297, Berlin, 1086; Tel. 2 09 32 48 (Dr. Grafik)

MODULA-2: PASCAL ohne die Nachteile von PASCAL

Steffen Hauptmann
Technische Universität Dresden, Sektion Informationstechnik, Bereich Automatisierungstechnik

Dipl.-Ing. Steffen Hauptmann (27) studierte von 1981–1986 an der Technischen Universität Dresden, Sektion Informationstechnik, Bereich Automatisierungstechnik. Seit Abschluß des Studiums ist er als Forschungsstudent am gleichen Wissenschaftsbereich beschäftigt.
Er arbeitet zur Zeit an Methoden zur Schaltung systemnaher, portabler Softwarekomponenten für die Automatisierungstechnik.

MODULA-2 wurde wie PASCAL von Prof. Niklaus Wirth an der Eidgenössischen Technischen Hochschule (ETH) Zürich entwickelt und ist PASCAL sehr ähnlich, stellt aber mehr als nur eine bloße Erweiterung dieser verbreiteten Programmiersprache dar.
Neu sind ein Modulkonzept, Möglichkeiten zur hardware- und betriebssystemnahen Programmierung, ein einfaches Koroutinenkonzept als Grundlage für die Implementierung paralleler Algorithmen sowie flexiblere String- und Feldverarbeitungsmöglichkeiten.
Der vorliegende Beitrag charakterisiert kurz die Ziele, die mit dem Einsatz von MODULA-2 verfolgt werden, gibt einen Überblick über die Geschichte dieser noch jungen Sprache und stellt Arbeiten vor, die mit dem Ziel durchgeführt wurden, MODULA-2-Entwicklungssysteme und eine einheitliche portable Modulbibliothek auf verschiedener in der DDR breit zur Verfügung stehender 8- und 16-Bit-Rechentechnik zu implementieren.

Motivation

In der DDR hat sich seit Beginn der achtziger Jahre die rechentechnische Basis stark verbreitert. Insbesondere die Zahl der eingesetzten Mini-, Mikro- und Personalcomputer wuchs rasch an. Qualitativ vollzieht sich die Entwicklung von der Klasse der 8-Bit- über die 16-Bit- zur 32-Bit-Rechentechnik. Heute existieren in der DDR alle drei Rechnerklassen nebeneinander, wobei die 8-Bit-Technik noch dominiert, der breite Übergang zur 16-Bit-Technik aber bereits vollzogen wird.

Auf dem Gebiet der Software ist die Entwicklung dadurch gekennzeichnet, daß selbst innerhalb einer Rechnerklasse viele unterschiedliche Betriebssysteme, Programmiersprachen und andere Softwarewerkzeuge Verwendung finden. Für den Softwareentwickler ergeben sich daraus große Probleme, zum einen bei der Entwicklung von Software auf parallel zur Verfügung stehender inhomogener Rechentechnik, zum anderen beim Übergang auf neu hinzukommende Rechnertypen.

Aus dieser Situation heraus entstand etwa 1984 der Wunsch, für eine Reihe weiterer Softwareprojekte eine einzige Programmiersprache als Grundlage einer einheitlichen Softwaretechnologie einzusetzen, wobei folgende Hauptforderungen so gut wie möglich erfüllt werden sollten:

- Die Verfügbarkeit der Sprache auf den in der DDR verbreiteten Rechnertypen mußte für die nächsten Jahre gesichert sein.
- Die Sprache sollte sich gleichermaßen gut für die System- und Anwendungsprogrammierung eignen, da beide Felder für die Automatisierungstechnik ähnlich wichtig sind. Sie sollte es ermöglichen, Funktionen, die normalerweise Bestandteil von speziellen Echtzeitsprachen bzw. Echtzeitbetriebssystemen sind, ohne größere Schwierigkeiten zu implementieren.
- Der Einsatz der Sprache sollte es ermöglichen, Software hochportabel zu gestalten, um zum einen auf unterschiedlichen Rechnern einheitliche Softwarekomponenten zur Verfügung zu haben und zum anderen mit geringem Aufwand den Übergang auf neu zu erwartende Rechner und Betriebssysteme zu ermöglichen.
- Die Sprache sollte modernen softwaretechnischen Anforderungen genügen, um eine möglichst hohe Produktivität der Softwareherstellung zu gewährleisten.

Spezielle Automatisierungssprachen wie PEARL oder RT-FORTRAN sind nicht verfügbar, traditionelle universelle Programmiersprachen wie BASIC, FORTRAN, COBOL oder PASCAL eignen sich in ihrer standardisierten Form nicht für die Systemprogrammierung, und spezielle Sprachdialekte wie TURBO-PASCAL widersprechen deutlich der Forderung nach hoher Portabilität.

Die sich besonders in der DDR zunehmend

OBERON – ein MODULA-2-Konkurrent?

Während MODULA-2 sich gerade anschickt, die Nachfolge von PASCAL anzutreten, erscheint OBERON – und will MODULA-2 diese Rolle streitig machen?
Niklaus Wirth hat sich wahrlich nicht auf seinen PASCAL-Lorbeeren ausgeruht. Er veröffentlichte (1987) in der Broschüre der ETH Zürich *From Modula to Oberon* eine Studie (innerhalb seiner Arbeiten an einem Betriebssystem) über wünschenswerte Erweiterungen zu MODULA. Danach beinhaltet OBERON ein Klassenkonzept (für objektorientierte Programmierung interessant) sowie zwei wesentliche Erweiterungen gegenüber MODULA-2:

- Typerweiterung (type extension)
- Typschachtelung (type inclusion).

Bei dieser OBERON-Studie ließ Wirth aber auch Dinge aus, die ihm „nicht vordringlich erscheinen und damit die Arbeit am Compiler vereinfachen". OBERON enthält keine Koroutinen (oder andere Ausdrucksmöglichkeiten für parallele Prozesse), Aufzählungstypen sowie Mengen. Damit stellt OBERON trotz seiner MODULA-Erweiterungen nur eine „Einschränkung (Untermenge) von MODULA" dar.
„Die Programmiersprache OBERON ist das Ergebnis einer konzentrierten Anstrengung, die Leistungsfähigkeit von MODULA-2 zu steigern und gleichzeitig die Komplexität zu verringern. Einige Elemente wurden weggelassen und einige wenige hinzugefügt, um die Ausdrucksfähigkeit und Flexibilität der Sprache zu erhöhen." Eine Veröffentlichung in der Zeitschrift *Software – Praxis and Experience"* ist angekündigt.

MP

Tafel 1 Vergleich mehrerer höherer Programmiersprachen (aus der Sicht des MODULA-2-Programmierers)

Eigenschaften		ADA	MODULA-2	PASCAL	PEARL	C	FORTRAN	PL/M	BASIC
Struktur-konzeption	Modulkonzept	+++	+++	–	++	–	–	+	
	Prozeduren, Funktionen	+++	+++	++	++	++	++	++	+
	Sequentielle Kontroll-strukturen	+++	+++	++	+++	+++	+	++	–
	Datenstrukturen	+++	+++	+++	++	++	+	++	–
	Datentypen, Operatoren	+++	++	++	++	+	+	+	+
Sicherheit gegen Programmierfehler		+++	+++	+++	++	+	–	+	–
Portabilität, Standardisierung		+++	++	+	++	++	+++	++	–
Lesbarkeit, Selbstdokumentation		+++	+++	+++	+++	+	–	+	–
Erlernbarkeit		+	+++	+++	++	+	++	++	+++
Ein-/Ausgabe-Unterstützung		–	–	+	+++	–	+++	–	+
Speicherverwaltung		+++	++	++	+	–	–	–	–
Echtzeitunterstützung		++	+	–	+++	–	–	–	–

– nicht unterstützt, schwierig handhabbar
\+ in Ansätzen vorhanden, als Ergänzung/Erweiterung unterstützt
++ unterstützt
+++ voll ausgebaut

größerer Popularität erfreuende Sprache C erfüllt alle obigen Forderungen einigermaßen gut, ist aber hinsichtlich der letzten beiden Punkte modernen, neu entwickelten Sprachkonzepten wie CHILL, ADA oder MODULA-2 deutlich unterlegen. ADA und CHILL zählen zu den sehr großen Sprachen (hinsichtlich des Sprachumfangs und folglich auch hinsichtlich der notwendigen Werkzeuge), stehen jedoch auf der in der DDR breit eingesetzten Rechentechnik nur sehr eingeschränkt zur Verfügung. Ganz anders dagegen die Sprache MODULA-2. Sie ist prinzipiell schon auf 8-Bit-Rechnern lauffähig und hat in kürzester Zeit international eine beträchtliche Resonanz und Verbreitung gefunden. Compilerimplementationen für alle verbreiteten Rechner- und Betriebssystemversionen liegen vor. Die Sprache gehört zum kleinen Kreis der vom Kombinat Robotron für seine zukünftige Produktpalette (ab 16-Bit-Rechner) unterstützten Sprachen.
Das war Anlaß dafür, MODULA-2 zur Grundlage einer Reihe von Softwareprojekten (u. a. für die Realisierung der Software eines Prozeßleitstandes) zu machen. Nun liegen erste Ergebnisse und dreijährige Erfahrungen beim Einsatz der Sprache vor. Diese sollen in die nachfolgenden Ausführungen mit einfließen.

Geschichtliches

Die Entstehung von PASCAL

Die Geschichte von MODULA-2 ist eng mit der von PASCAL verknüpft. Beide Sprachen stammen von Niklaus Wirth, Professor an der ETH Zürich /1/, /2/.
Wirth gehörte bereits in den sechziger Jahren zu einer Gruppe von Sprachexperten, die eine Nachfolgeversion von ALGOL-60 erarbeiten sollte. Zusammen mit Hoare erarbeitete er 1966 den Sprachvorschlag ALGOL-W, der sich aber innerhalb der Expertengruppe nicht durchsetzen konnte. Die Sprache war einfach strukturiert, systematisch aufgebaut und forderte die konsequente Deklaration aller Programmvoraussetzungen. Niklaus Wirth erkannte, daß sich Sprachentwürfe nur schlecht gegen die vielschichtigen Anschauungen einer Expertengruppe durchsetzen ließen, und er beschloß, eine eigene Sprache als Hilfsmittel für die Lehre und Forschung an der ETH Zürich zu entwickeln. Diese Sprache, die er PASCAL nannte, enthielt neuartige Konzepte, die es erlaubten, Daten ähnlich wie Algorithmen zu strukturieren. Sie war vollständig maschinenunabhängig und relativ leicht auf Mikrorechnern zu implementieren.
PASCAL war eine Hochschulentwicklung und hatte, im Gegensatz zu den meisten anderen verbreiteten Programmiersprachen, anfangs keinerlei kommerzielle Unterstützung durch Rechnerhersteller. Daß sie sich trotzdem innerhalb kurzer Zeit international breit durchsetzte, spricht für ihre hervorragende Qualität.

Ein neues Projekt

Trotz eines Angebotes zur Mitwirkung an der Entwicklung der „Komitee"-Sprache ADA entschloß sich Wirth 1977, selbst ein kombiniertes Hard- und Softwareprojekt zur Entwicklung eines Rechners, der von Hardware bis zur Anwendersoftware ausschließlich in einer Sprache programmiert werden sollte, ins Leben zu rufen /3/.
PASCAL schien für diesen Zweck wenig geeignet, auch wollte Wirth nicht einfach einen weiteren Dialekt der Sprache PASCAL entwickeln und damit ein rundes Konzept durch eine aufgesetzte Erweiterung verunstalten. So entstand die neue Sprache MODULA-2. Diesen Namen erhielt die Sprache, weil sie einer MODULA genannten Sprache sehr nahesteht, die Wirth zwei Jahre zuvor für Multiprogramming-Experimente entwickelt hatte /4/. Der erste MODULA-2-Compiler entstand für einen PDP-11 von Digital Equipment Corporation (DEC) /5/. Es war ein 5-Pass-Compiler, der die Voraussetzung zur Softwareentwicklung für den Rechner Lilith darstellte. Der Lilith selbst war die Hardware-Komponente des MODULA-2-Projekts /3/. Er ist aus Bit-Slice-Prozessoren aufgebaut und voll auf die Programmierung mittels MODULA-2 ausgerichtet. Seine Maschinensprache ist der sogenannte M-Code. Für diesen Rechner wurde der zweite MODULA-Compiler entwikkelt, ein 4-Pass-Compiler /6/.
1986 wurde der Lilith-Nachfolgerechner Ceres gebaut, der einen 32-Bit-Mikroprozessor NS320032 von National Semiconductor enthält. Der Rechner erreicht die Leistungsfähigkeit der VAX 11-780 von DEC bei wesentlich geringerem Stückpreis /7/.
Gleichzeitig wurde ein neuer Markstein für MODULA-2-Compiler-Implementationen geschaffen, ein Single-Pass-Compiler, der um ein Mehrfaches schneller ist als ein Mehr-Pass-Compiler.

Die internationale Resonanz

Schon kurz nach den ersten Veröffentlichungen zu MODULA-2 entstand international ein reges Interesse an dieser Sprache, das wohl auf die enge Verwandtschaft mit PASCAL zurückzuführen war. „PASCAL ohne die Nachteile von PASCAL" charakterisierte Jerry Pournelle, Kolumnist der Zeitschrift *Byte* die Sprache recht treffend /7/.
Ausgehend vom Ur-Compiler entstanden MODULA-2-Implementationen für viele Rechner, von CP/M-80-Systemen über MS-DOS-Varianten bis hin zu Mainfraimes. Zusätzlich entstanden ganz neue, sehr leistungsfähige Konzepte, wie das inkrementelle Compilerprojekt MODULA-2 Software Development System (M2SDS) von Interface Technologie für MS-DOS und andere Betriebssysteme, das die Übersetzungsgeschwindigkeit von TURBO-PASCAL erreicht.
Renommierte Informatiker billigen MODULA-2 gute Chancen im internationalen Ringen mit dem „Riesenbaby" ADA zu, nicht zuletzt deshalb, weil viele Konzepte beider Sprachen ähnlich leistungsfähig sind, ein ADA-Compiler aber etwa die zehnfache Größe eines MODULA-2-Compilers erreicht /8/.
Für die Entwicklung von MODULA-2 erhielt

Prof. Wirth neben vielen anderen Ehrungen 1984 die IEEE-Auszeichnung des amerikanischen Instituts of Electrical and Electronic Engineering, so etwas wie den Nobelpreis der Branche.

Eigene Arbeiten

Etwa um 1984 gab es in der DDR erste Arbeiten an und mit MODULA-2-Entwicklungssystemen durch die Humboldt-Universität Berlin (HUB), die auf den beiden ersten Urcompilern für den PDP-11 und den Lilith aufbauten. Dabei war der PDP-11-Compiler vom Betriebssystem LAOS (RT11) auf MOOS1600 (RSX11M) auf K-1630-Rechner umgesetzt worden. Für 8-Bit-Bürocomputer wurde ein M-Code-Interpreter geschaffen, der es ermöglichte, den Lilith-Compiler für diese Rechner einzusetzen.
Die HUB-MODULA-2-Systeme wählten wir als Ausgangspunkt für erste eigene Programmierversuche. Schnell reifte die Erkenntnis, daß diese Arbeitsversionen zwar für erste Experimente geeignet waren, aber noch eine Reihe von Eigenheiten besaßen, die sie für unsere Nutzungsziele ungeeignet machten. Beide Systeme stützten sich nach wie vor auf eine RT11-orientierte Betriebssystemschnittstelle. Das brachte insbesondere unter RSX11M im Mehrnutzerbetrieb erhebliche Probleme im E/A-Zeitverhalten mit sich, so daß viele Nutzer die Anwendung des Systems als nicht akzeptabel ansahen. Zudem existierte keine leistungsfähige, systematisch geordnete und gut dokumentierte Modulbibliothek, was die Nutzung der Sprache stark erschwerte.
Die 8-Bit-Systeme unter SCP und UDOS waren stark betriebssystemabhängig, ja sie erforderten sogar speziell dafür generierte Betriebssystemversionen. Damit liefen sie nur auf wenigen Rechnern.
Da die Humboldt-Universität nach eigenen Aussagen von vornherein nicht das Ziel verfolgte, diese Implementationen über den Rahmen von Arbeitsversionen hinaus weiterzuentwickeln, beschlossen wir, als ersten Schritt unserer Arbeit die Leistungsfähigkeit der Entwicklungssysteme zu verbessern und zu erweitern und damit die Voraussetzungen für weitere Softwareprojekte unter Nutzung der Sprache MODULA-2 zu schaffen.

Eine portable Modulbibliothek als Entwicklungsgrundlage

Eine Grundlage für die Neugestaltung der Systeme bildete die Konzeption einer für alle Rechner möglichst einheitlichen leistungsfähigen Modulbibliothek /9/. Diese Bibliothek ist strukturell geteilt in einen kleineren betriebssystemspezifischen Teil, der es gestattet, die Leistungen des unterliegenden Betriebssystems bzw. der Hardware voll auszuschöpfen und in einen wesentlich größeren, portabel gestalteten Teil, der auf allen Rechnern in gleicher Weise implementiert wird und damit eine Voraussetzung für hochportable Software bildet.
Die Portierung der Bibliothek erfordert lediglich den Austausch des betriebssystemspezifischen Teils. Bisher liegen Versionen für CP/M-80 (CP/A), SCP, UDOS, RSX11M (MOOS1600, OS-RW), MS-DOS (PC-DOS), DCP sowie zum Teil für 8-Bit-OEM-Systeme ohne Betriebssystem vor.
Die Compiler für RSX11M und die 8-Bit-Systeme wurden dieser neuen Bibliothek angepaßt. Der M-Code-Interpreter für die 8-Bit-Systeme wurde völlig betriebssystemunabhängig gestaltet. Ein großer Teil der Entwicklungs- und Testarbeiten konnte dabei auf leistungsfähigen PCs unter MS-DOS erfolgen, da die Bibliothek auch in dieses Betriebssystem portiert wurde und hier mehrere leistungsfähige MODULA-Systeme als Entwicklungswerkzeuge zur Verfügung standen. Nach Abschluß der Entwicklungsarbeiten am 8-Bit-System erfolgte die Rückportierung auf die entsprechenden 8-Bit-Rechner.
Die Nutzung der 8-Bit-MODULA-Systeme ist auf Grund des verwendeten M-Code-Interpreter-Systems als Kompromiß hinsichtlich der frühzeitigen Vorbereitung auf leistungsfähigere 16-Bit-Rechentechnik anzusehen. Die Übersetzungszeiten des 4-Pass-Compilers liegen relativ hoch, aber er gestattet die Nutzung von MODULA-2 ohne Einschränkungen auf Rechnern, die in der DDR in großer Zahl vorhanden sind und wird in einer integrierten Umgebung mit vielen weiteren Tools sowie einer leistungsfähigen Bibliothek angeboten.
Die Zeitnachteile sind bis zu einem gewissen Grad dadurch kompensierbar, daß problemlos neue Bibliothekskomponenten und Programme, die den portablen Teil der Bibliothek nutzen, von anderen Rechnern übernommen werden können. Der Übergang zwischen zwei Rechnersystemen ist mit geringem Aufwand vollziehbar. Eine vollständige Cross-Entwicklung von Software für 8-Bit-Systeme unter MS-DOS ist möglich.

Die neuen Konzepte von MODULA-2
Das Modulkonzept

Bereits der Name MODULA-2 verrät es, das Modulkonzept ist das wichtigste neue Konzept der Sprache. Es geht auf Parnas zurück /10/. Was im Maschinenbau und anderen konstruktiven Bereichen praktisch nicht mehr wegzudenken ist, hält nun auch in der Softwaretechnologie Einzug: Große Systeme werden aus einzelnen, relativ unabhängigen und austauschbaren Bestandteilen, den Modulen, zusammengesetzt. Dieser Gedanke hielt Einzug in eine Anfang der achtziger Jahre entstandene Generation neuer Programmiersprachen, wie MESA, MODULA-2, CHILL und ADA.
Das Modulkonzept ermöglicht bzw. stimuliert die Realisierung einer ganzen Reihe weiterer Konzepte. Modularisierung ist eine zusätzliche Form der Programmstrukturierung mit weitreichenden Effekten. Dabei ist es durchaus nicht von vornherein klar, welche Teile eines Programms in einem Modul zusammengefaßt werden sollen. Denkbar sind u. a.:

- Sammlungen von Funktionsprozeduren ähnlicher Art (z. B. mathematische Standardfunktionen oder Konvertierungsroutinen)
- Bereitstellung eines (in seiner Struktur unsichtbaren) Datentyps und aller darüber möglichen Operationen (Datenkapselung, Datenabstraktion, z. B. Listenverwaltung oder Geräte-E/A)
- Zusammenfassung programmweit genutzter Definitionen (z. B. Fehlerkodes oder ASCII-Zeichensatz)
- Abgrenzung beliebiger, von vielen Programmen genutzter Algorithmen in Bibliotheksmodulen (z. B. Filenameneingabe und -zerlegung).

Übergreifend gilt die Regel, daß die Modularisierung um so besser ist, je kleiner die Modulschnittstelle und je größer der innere Zusammenhalt der im Modul realisierten Programmteile ist.
Alle Leistungen, die ein Modul seiner Umgebung zur Verfügung stellt, werden mit Hilfe einer Exportschnittstelle syntaktisch exakt beschrieben. Diese Exportschnittstelle ist bei Bibliotheksmodulen, d. h. Modulen, die von mehreren Programmen genutzt werden sollen, physisch von der Modulimplementation getrennt. Zusammen mit Kommentaren, die die Semantik der Modulschnittstelle ausreichend beschreiben, kann diese Exportschnittstelle dem Nutzer eine hinreichende Dokumentation zur Nutzung des Moduls zur Verfügung stellen. Die Implementation ist austauschbar, ohne daß andere Teile des Programmsystems davon berührt werden.
Die abgetrennte Exportschnittstelle eines Moduls nennt man Definitionsteil eines Moduls oder kurz Definitionsmodul. Die Trennung von Definition und Implementation eines Moduls ist besonders interessant für die Teamarbeit an größeren Programmsystemen. Zentral oder von jedem Teammitglied werden die Definitionsteile der von ihm zu implementierenden Module festgeschrieben. Damit kennt jedes Teammitglied exakt die Struktur des Programmsystems und damit die Leistungen, auf die es aufbauen kann, ohne daß bereits eine Programmzeile implementiert wurde. Nun ist eine weitgehend unabhängige Implementierung und Testung der Module möglich.

Portable Software

Ganz allgemein betrachtet steht einem Programmierer für die Lösung seiner Aufgabe ein Satz von Funktionen zur Verfügung, mit denen er den Zustand (z. B. Speicherbelegung, E/A-Geräte) seines Rechners beeinflussen kann.
Funktionen und Zustand bilden eine Art virtuelle Maschine, mit der er arbeitet. Im einfachsten Fall ist das die nackte Hardware, in der Regel stehen ihm aber weitere (komplexere) Funktionen, z. B. ein vorhandenes Betriebssystem, die verwendete Programmiersprache oder nutzbare Bibliotheken, zur Verfügung. Es existiert eine Hierarchie virtueller Maschinen unterschiedlicher Leistungfähigkeit. Soll Software portabel gestaltet werden, so muß sie irgendwo auf eine virtuelle Maschine aufbauen, die auf mehreren Rechnern (den Zielrechnern für die Portierung) in gleicher Form zur Verfügung steht.
In MODULA-2 kann dem Programmierer eine Hierarchie solcher virtuellen Maschinen unterschiedlicher Leistungsfähigkeit und damit unterschiedlicher Portierbarkeit durch ein hierarchisches System aufeinander aufbauender Module zur Verfügung gestellt werden.
Soll Software zwischen mehreren Rechnern portiert werden, dann wird die vom Niveau her am tiefsten liegende und damit am einfachsten zu implementierende virtuelle Maschine definiert, die auf allen diesen Rechnern realisierbar ist. Alle Programmteile, die diese Maschine verwenden, sind portabel, alle die Teile, die diese Maschine selbst realisieren, müssen neu implementiert werden.
Das Modulkonzept ermöglicht es, auf sehr flexible Art und Weise das Prinzip der virtuellen Maschine zur Erstellung portabler Software zu implementieren und zu nutzen. Zum einen stellen die Definitionsmodule eine natürliche Schnittstelle einer oder eines Teils einer virtuellen Maschine für den Anwender dar, zum anderen lassen sich leicht Modulhierarchien schaffen, bei denen komplexere virtuelle Maschinen auf einfacheren aufbauen.

Maschinennahe Programmierung

Eine Systemprogrammiersprache muß die Möglichkeit bieten, hardware- und betriebssystemnah zu programmieren, das heißt, über Adressen auf den Hauptspeicher zuzugreifen, E/A-Ports zu programmieren, Betriebssystemrufe zu verwenden, unter Umständen auch Maschinenkode direkt in ein Programm einzufügen.
Diese Möglichkeiten stellt MODULA-2 dem Programmierer mittels des Moduls SYSTEM zur Verfügung. Hervorzuheben ist, daß hier auch die Datentypen WORD und ADDRESS zur Verfügung gestellt werden. Als formale Parameter einer Prozedur sind WORD mit jedem skalaren Datentyp gleicher Größe, ADDRESS mit jedem Zeigertyp und ARRAY OF WORD mit jedem beliebigen Typ kompatibel. Das ermöglicht die Realisierung von Prozeduren mit typenunabhängigen aktuellen Parametern.
Weiterhin bietet MODULA-2 die Möglichkeit, eine kontrollierte Typumwandlung zwischen Typen gleicher Speichergröße durchzuführen, wobei der Zieltyp wie eine Art Prozedurbezeichner genutzt wird, z. B.

```
VAR c: CHAR; b: BOOLEAN;
...
c := CHAR(b); b := BOOLEAN(c);
```

Das Koroutinenkonzept

MODULA-2 bietet dem Anwender zwar keine Objekte im Sinne von Tasks an, aber elementare Konstruktionen zur Realisierung derartiger Objekte sind vorhanden. Jedes MODULA-Programm stellt einen Prozeß (eine sogenannte Koroutine) dar, und mittels der Anweisung *NEWPROCESS* können beliebige parameterlose Prozeduren als weitere Prozesse deklariert werden.
Ein Wechsel zwischen zwei Prozessen kann explizit mittels der *TRANSFER*-Anweisung erfolgen oder über die *IOTRANSFER-Anweisung* an einen Interrupt gekettet werden. Der Prozeß, der die Kontrolle über die CPU erhält, wird genau an der Stelle fortgesetzt, an der er zuletzt seine Kontrolle abgegeben hat. Aufbauend auf diesen drei elementaren Sprachkonstruktionen ist es möglich, Taskstrukturen zu definieren, beliebige Schedulingalgorithmen zu implementieren und Module für die Synchronisation, Kommunikation usw. zur Verfügung zu stellen, also Leistungen, die normalerweise von Echtzeitbetriebssystemen angeboten werden. Alternative Algorithmen (besonders schnelle, sehr komfortable usw.) können wahlweise zur Verfügung gestellt werden.

Offene Feldgrenzen für Prozedurparameter

Prozeduren, die z. B. Strings oder Real-Felder verarbeiten, sollten dies oft zweckmäßig für Felder beliebiger Länge realisieren. Das ist in MODULA-2 möglich. Felder können als formale Prozedurparameter ohne Feldgrenzen deklariert werden. Innerhalb der Prozedur kann man mittels der Funktion HIGH die aktuelle Länge des Feldes bestimmen.

Ausblick

MODULA-2 ist eine Sprache, die auf den bewährten, international anerkannten Konzepten von PASCAL aufbaut und diese konsequent weiterentwickelt. Gegenüber älteren Programmiersprachen wie FORTRAN, PASCAL, aber auch C bietet MODULA-2 eine ganze Reihe neuer softwaretechnischer Konzepte, die die Sprache zu einem leistungsfähigen Werkzeug für breite Gebiete der System- und Anwenderprogrammierung machen. Praktisch realisierte Softwaresysteme in MODULA-2, wie das Datenbanksystem LIDAS /11/, der grafische Dokumenteneditor ANDRA /12/, das fehlertolerante, verteilte Echtzeitbetriebssystem FUTURE /13/ und vieles andere, belegen das deutlich.
Überall dort, wo keine pragmatischen Gründe zum Festhalten an traditionellen Sprachen zwingen oder wo die Schranken dieser Sprachen einer leistungsfähigen Softwaretechnologie und damit letztlich niedrigen Softwarekosten entgegenstehen, setzt sich MODULA-2 zunehmend durch /14/.

Literatur

/1/ Wirth, N.: Revidierter Bericht über die Programmiersprache Pascal. Akademie-Verlag, Berlin 1976
/2/ Wirth, N.: Programming in Modula-2. Springer-Verlag, Berlin, Heidelberg, New York 1985
/3/ Wirth, N.: The personal computer Lilith. Bericht Nr. 40 der ETH Zürich, Institut für Informatik, 1981
/4/ Wirth, N.: Modula: a language for modular multiprogramming. Software Practice and Experience, Vol. 7, 1977
/5/ Geissmann, L.: Overview of the Modula-2 Compiler M2RT11. Rechnerausdruck, Institut für Informatik der ETH Zürich, 1981
/6/ Geissmann, L.; Jacobi, Ch.: Overview of the Modula-2 Compiler M2M. Rechnerausdruck, Institut für Informatik der ETH Zürich, 1981
/7/ Henger, G.: Niklaus Wirth, der Schöpfer von Modula-2: „Gute Sprachen kommen aus einem einzigen Kopf". Chip Plus (1987) 8
/8/ Rechenberg, P.: Ada und Modula-2 – Programmiersprachen der achtziger und neunziger Jahre? Zeitschrift der Johannes-Kepler-Universität Linz (1983) 4
/9/ Hauptmann, S.: Eine portable Modula-2-Systembibliothek. Internes Arbeitsmaterial, TU Dresden, Sektion Informationstechnik, Bereich Automatisierungstechnik, 1988
/10/ Parnas, D. L.: On the criteria to be used in decomposing systems into modules. Communications of the ACM, Nr. 15, Dezember 1972, S. 1053
/11/ Pomberger, G.: Lilith und Modula-2: Werkzeuge der Softwaretechnik. Hanser-Verlag, München 1985
/12/ Gutknecht, J.; Winiger, W.: Andra: The Document Preparation System of the Personal Workstation Lilith. Software Practice and Experience, Vol. 14, 1984
/13/ Demmelmeier, F.: Betriebssystemfunktionen im fehlertoleranten Multimikrocomputersystem FUTURE. rtp (1984) 9, S. 408
/14/ Lindner, U.; Trautloft, R.: Grundlagen der problemorientierten Programmentwicklung. VEB Verlag Technik, Berlin 1987

✉ KONTAKT ☎

Technische Universität Dresden, Sektion Informationstechnik, Bereich Automatisierungstechnik, Mommsenstraße 13, Dresden, 8027; Tel. 4 63 50 70

MC 68010 im Überblick

Wolf-Dietram Bretschneider

Es werden der Aufbau und die grundlegende Arbeitsweise des 16-Bit-Mikroprozessors MC 68010 von Motorola behandelt. Dabei werden Besonderheiten im Registersatz, Businterface und speziellen Befehlsausführungen genauer erläutert. Ausführlich wird auf die Anschlußbelegung und die Funktion der einzelnen Leitungen der CPU eingegangen. Unterschiede zum Vorgängermodell MC 68000 sind im Text hervorgehoben. Außerdem stellt eine Übersicht die übrigen 16- bzw. 32-Bit-Prozessoren von Motorola vor.

Allgemeines

Der 68000 von Motorola stand am Anfang der Entwicklung einer Prozessorfamilie, die von der oberen Grenze der 16-Bit-Technik bis zum echten 32-Bit-Mikroprozessor reicht. Die internen Universalregister der CPUs deuteten von Beginn an auf entsprechende Nachfolger hin. Die derzeitige 68000-Familie wird in Tafel 1 vorgestellt.
Die folgenden Ausführungen gelten für den 68010 als pinkompatible Weiterentwicklung des 68000, und etwaige Unterschiede werden im Text vermerkt. Die Anschlußbelegung und das Funktionsschaltbild stellt Bild 1 dar.

Anschlußbeschreibungen

Der Speicher ist byteorientiert, und dessen Adressen liegen am Adreßbus A1 . . . A23 an. Die Auswahl des entsprechenden Bytes und damit die benutzte Übertragungsbreite des Datenbusses D0 . . . D15 erfolgt mit den Leitungen $\overline{LDS}$ (unteres Byte) und $\overline{UDS}$ (oberes Byte) innerhalb eines Zugriffszyklus. Der Datenbus überträgt Wortinformationen bei gleichzeitigem Aktivieren von $\overline{LDS}$ und $\overline{UDS}$. Die Bussteuerung kann beim 68010 sowohl asynchron als auch zum Zwecke der Weiterverwendung der MC 6800-Peripherie synchron erfolgen. Der asynchrone Bus erlaubt die Verwendung von Peripherie mit unterschiedlichen Zugriffszeiten. Dabei gibt der Ausgang $\overline{AS}$ an, wann sich eine gültige Adresse am Adreßbus befindet. Am Eingang $\overline{DTACK}$ wird der Transfer mit Peripherie oder Speicher gesteuert. Der Prozessor führt Wartezyklen aus, bis $\overline{DTACK}$ aktiv ist. Bei einem Lesezugriff werden die Daten mit $\overline{DTACK}$ = 0 von der CPU übernommen. Bei einem Schreibzugriff liefert die CPU die Daten bis

***Tafel 1** Übersicht der Motorolaprozessoren*

Typ	Merkmale
68000	*Erster Prozessor mit internen 32-Bit-Register, 16-Bit-Daten- und 24-Bit-Adreßbus, maximal 12,5 MHz Takt.*
68008	*Abgerüstete Variante mit 8-Bit-Datenbus, 20-Bit-Adreßbus mit A0-Leitung statt $\overline{LDS}$ und $\overline{UDS}$, 4 Interruptebenen, da nur zwei Eingänge, kein Ausgang $\overline{VMA}$ bei synchronem Buszugriff, intern wie 68000.*
68010	*Weiterentwicklung des 68000 mit Vektor-Basisregister und zwei Alternativ-Coderegistern, erweiterter Befehlssatz, 16-Bit-Daten- und 24-Bit-Adreßbus, veränderte Vektorentabelle.*
68012	*Erweiterung des verfügbaren Adreßraumes vom 68010 durch die Ausgänge A24 . . . A29 und A31, A30 nur intern, 16-Bit-Daten- und 31-Bit-Adreßbus.*
68020	*32-Bit-Prozessor mit zwei zusätzlichen Registern für den integrierten 256-Byte-Befehlscache, 32-Bit-Daten- und Adreßbus, Bitfeld- und ungepackte BCD-Operandeu, 3fach Adreßregister A7 als Stackpointer, maximal 16 MHz Takt.*
68030	*Weiterentwicklung des 68020 mit je 256 Byte Cache für Befehle und Daten, intern doppelt ausgelegte Bussysteme, integrierte MMU, maximal 24 MHz Takt.*

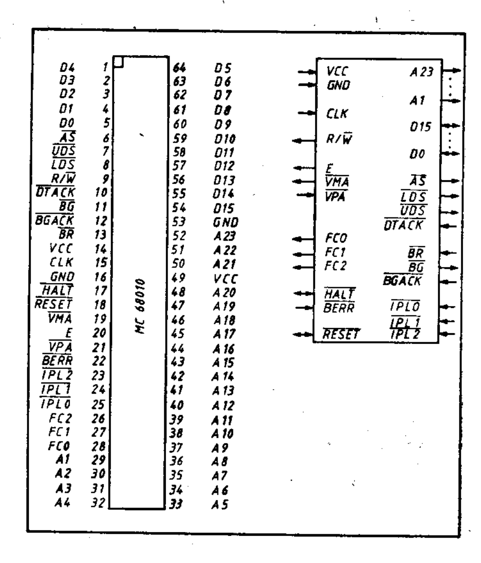


Bild 1 Anschlußbelegung und Schaltzeichen

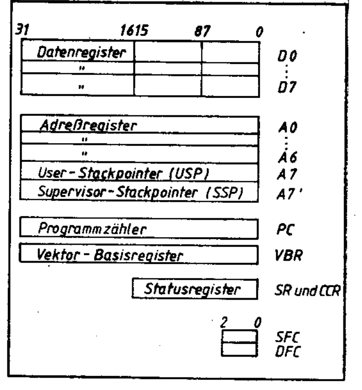


Bild 2 Supervisor-Programmiermodell

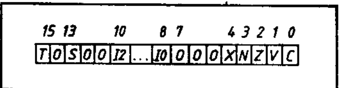


Bild 3 Statusregister des 68010

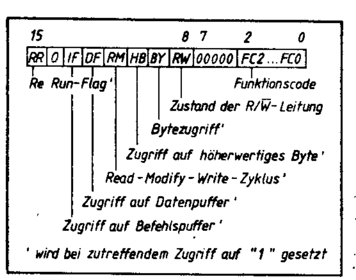


Bild 4 Spezialstatuswort bei Bus- oder Adreßfehlern

DTACK = 0 ist. Die Art der Zugriffe unterscheidet man am Ausgang R/W. Außerdem signalisiert DTACK im Interruptanerkennungszyklus, daß ein Vektor der auslösenden Einheit an D0 ... D7 anliegt. Der synchrone Bus wird durch die Signale VPA, E und VMA realisiert. Dabei wird an VPA der synchrone Buszugriff angefordert und mittels VMA der Peripherie mitgeteilt, daß die nächste Aktivierung von Ausgang E (High) einen synchronen Zugriff einleitet. Die Leitung E liefert den durch Zehn geteilten Systemtakt (sechs Systemtaktzyklen Low und vier Zyklen High). Der Synchrontakt muß zur Gewährleistung der Arbeit der Peripheriebausteine freilaufend sein. Die volle Geschwindigkeit des 68010 kann in dieser Betriebsart nicht erreicht werden. Der Eingang VPA wird ebenfalls aktiviert bei Auslösung eines Autovektorinterrupts. Hier erzeugt der 68010 selbständig die entsprechende Vektornummer zum Sprung in die Interruptroutine. Zur externen Interruptauslösung stehen die Pins IPL0 ... IPL2 zur Verfügung, deren statisch anliegende 3-Bit-Information die Prioritätsstufe darstellen und nach Vergleich mit der Interruptmaske im Statusregister eine Interruptanerkennung zur Folge haben kann. Die dazugehörige Quittung wird neben anderen Statusanzeigen an den Funktions-Code-Leitungen FC0 ... FC2 angezeigt. Die Realisierung der Busübergabe an DMA-Bausteine bzw. an andere CPUs in Multiprozessorsystemen wird durch drei Anschlüsse unterstützt. Die Busanforderung signalisiert BR, und mit BG erfolgt die Bestätigung der Busfreigabe. Der Eingang BGACK zeigt dem Prozessor an, daß eine andere Einheit den Rechnerbus übernommen hat. Von der CPU wird die Bestätigung der Busübergabe erst nach aktivem AS- und inaktivem DTACK-Signal gegeben, und bevor AS nicht wieder *High* ist, sollte BGACK noch nicht *Low* werden. Die übrigen drei Anschlüsse BERR, RESET und HALT dienen der Systemsteuerung. Dabei meldet BERR dem Prozessor einen fehlerhaften Zustand des Busses. Falls BERR und HALT gleichzeitig aktiv werden, erfolgt eine Wiederholung des laufenden Zyklusses (Re-Run-Zyklus). Der bidirektionale Anschluß HALT bewirkt außerdem das Stoppen der CPU am Ende des gerade laufenden Buszyklusses. Als Ausgang meldet er das Auftreten eines Doppelfehlers, das heißt, bei erkanntem Bus- oder Adreßfehler tritt vor Abarbeitung des ersten Befehls der Unterbrechungsroutine dieser Fehler wieder auf. Der RESET-Anschluß ist ebenfalls bidirektional und zeigt als Ausgang die Abarbeitung des Resetbefehls an. Der 68010 wartet in diesem Fall 124 Takte, bis er mit dem nächsten Befehl fortsetzt. So können softwaremäßig Peripheriebausteine rückgesetzt werden. Beim Rücksetzen der CPU mittels des Reseteinganges wird im Statusregister der Supervisormode eingeschaltet und die Interruptmaske Ebene 7 = 111 (nur NMI) geladen. Das Vektor-Basisregister wird für den definierten Systemstart mit Null geladen. Die untersten vier Bytes im Programmspeicher enthalten *die* Werte des Systemstackpointers und des Programmzählers, die zum erneuten Beginn nötig sind.

Prozessoraufbau

Das Bild 2 zeigt die interne Registerstruktur des 68010. Alle Adreß- und Datenregister lassen sich auch als Indexregister einsetzen. Durch die Unterteilung in Supervisor- und Usermode existieren auch zwei dazugehörige Stackpointer, wozu das doppelt vorhandene Adreßregister A7 und A7' verwendet wird. Vom Programmzähler werden nur die unteren 24 Bit benötigt. Das Statusregister stellt Bild 3 gesondert vor. Es unterteilt sich in Anwender(CCR)- und System(SR)-Byte. Letzteres läßt sich im Gegensatz zum 68000 bei den Nachfolgern nur im Supervisormode lesen. Im folgenden soll das Statusregister erläutert werden.

Anwenderbyte:

Bit 0 (C) Übertragsflag
Bit 1 (V) Überlaufflag zur Anzeige des überschrittenen Zahlenbereichs
Bit 2 (Z) Nullflag
Bit 3 (N) Vorzeichenflag
Bit 4 (X) Erweiterungsflag, verhält sich wie das C-Flag, jedoch nicht bei allen Befehlen. Der Zustand des Carryflags bleibt über die folgende Operation erhalten.

Systembyte:

Bits 8, 9, 10 Interruptmaske, die nur dann eine Unterbrechung zuläßt, wenn die Prioritätsstufe an IPL0 ... IPL2 höher ist.
Bit 13 Supervisorstatus wenn High und Userstatus mit beschränkter Ausführbarkeit der Befehle wenn Low.
Bit 15 Tracemodus, das heißt, nach jeder Befehlsabarbeitung ist eine Ausnahmeverarbeitung angeschlossen (Schrittbetrieb)

Im folgenden sollen die drei Register beschrieben werden, die der ursprüngliche MC 68000 nicht besaß. Das Vektor-Basisregister VBR gibt den Anfang der Vektorentabelle im Speicher an und ermöglicht damit die Verwendung alternativer Tabellen. Beim 68000 mußte die Vektorentabelle am Beginn des Speicherbereiches liegen. Die zwei Register DSC (Destination-Function-Code) und SFC (Source-Function-Code) lassen Zugriffe in fremde Speicherbereiche zu, da der Inhalt bei der Abarbeitung spezieller Befehle an den Funktionscodeausgängen der CPU ausgegeben wird. Es sind Kombinationen zu wählen, die bei der normalen Programmbearbeitung nie an den FC-Ausgängen auftreten. Dabei wird das SFC-Register für Lesezugriffe und das DFC-Register zum Schreiben in fremden Bereichen benutzt.

Programmbearbeitung

Es gibt grundsätzlich zwei Arbeitszustände, wenn man den Haltzustand nicht als solchen betrachtet. Für den normalen Programmablauf existieren die Grundregeln, daß auf Worte oder Langworte nur von geraden Adressen aus zugegriffen werden kann, während auf Bytes sowohl von geraden als auch ungeraden Adressen aus zugegriffen wird. Bestimmte Befehle lassen Besonderheiten im Programmablauf zu. So können Daten in die CPU eingegeben, verändert und anschließend in die gleiche Adresse zurückgeschrieben werden. Dieser sogenannte Read-Modify-Write-Zyklus verändert kein Prozessorregister und kann infolge kurzzeitig inaktiven AS-Ausgangs von einer Busanforderung unterbrochen werden. Beim Test-und-Setze-Befehl (TAS) ist diese Unterbrechung nicht möglich, und dieser Befehl schafft die Voraussetzung von Semaphor-Operationen. Semaphore in der Betriebssystemtechnik sind Signale, die Kenntnis über Prozeßzugriffe geben können.

Neben dieser Art der Programmbearbeitung existieren Befehle, die in Verbindung mit dem Befehl „DB cc" (Springe relativ, falls Bedingung erfüllt) den sogenannten Loop-Modus zulassen, der vom Prozessor selbständig eingenommen wird. Allerdings darf die Sprungweite innerhalb der Schleifen aufgrund der vorhandenen Register für Befehldekodierung und Prefetch (Laden von Befehlen während der Bearbeitung vorangegangener) nur vier Bytes betragen. Die Schleifendurchläufe erfordern keine Befehlsladezyklen mehr.

Die Unterbrechungsstruktur läßt genügend Möglichkeiten einer Ausnahmebehandlung zu. Dabei müssen Unterschiede zum 68000 beachtet werden. Da die Kombination 111 an

den FC-Leitungen einen CPU-Spacezyklus anzeigt, muß eine Kontrolle der Adreßbits A16 bis A19 erfolgen, die im Falle eines Breakpoints Low und bei der Interruptquittung High sind. Beim 68000 kann kein Breakpoint gesetzt werden, und die genannte Kombination gilt immer als Interruptquittung. Die Verarbeitung von Ausnahmen (Exceptions) wurde beim 68010 verfeinert, indem mehr Daten auf dem Stack abgelegt und damit auch virtuelle Rechnerkonzepte ermöglicht werden können. Die Ausnahmefälle sind durchnumeriert, und die CPU wählt den zugehörigen Vektor aus. Zu Beginn der Exceptionsverarbeitung werden das Statusregister und die Rücksprungadresse gerettet, das Supervisorbit gesetzt und das Tracebit rückgesetzt. Bei extern ausgelöstem Interrupt erhält die Maske den entsprechenden Wert, um ein Sperren niedriger priorisierter Interrupts zu veranlassen. Die Prioritäten und den Verarbeitungsbeginn erläutert Tafel 2. Die Ausführung des Befehls RTE (Rückkehr vom Interrupt) bringt eine Reihe von Tests zum ordnungsgemäßen Wiedereintritt ins Programm mit sich.
Dazu wird am Beginn der Ausnahmebehandlung nach dem Inhalt des Statusregisters und Programmzählerstand noch ein viertes Wort

Tafel 2 Unterbrechungsprioritäten und Verarbeitungsbeginn

Gruppe[1]	Rang[2]	Bezeichnung	Verarbeitungsbeginn
0	0	Reset	innerhalb
0	1	Busfehler	von zwei
0	2	Adreßfehler	Taktzyklen
1	0	Trace-Modus	vor dem
1	1	Interrupt	nächsten Befehl
2	0	illegaler Befehl	am Ende eines
2	1	nicht implementierter Befehl	Buszyklusses
2	2	Privilegverletzung	
3	0	Trapbefehle	nach dem aus-
3	1	Formatfehler	lösenden Befehl
3	2	Division durch Null	

[1] Prioritätsgruppe null ist die höchste
[2] Prioritätsverteilung innerhalb der Gruppe

gerettet (entspricht kurzem Stackformat). Es enthält die um zwei Bit verschobene Vektornummer der Exception in den unteren 12 Bits. Die restlichen 4 Bit kennzeichnen das Stackformat, weil bei Unterbrechungen der Gruppe *null* ein aus 29 Worten bestehendes, langes Format verwendet wird. Damit realisiert man die Speicherung des gesamten internen Prozessorzustandes und die infolge des Prefetch schwieriger gewordene Programmfortsetzung nach Bus- oder Adreßfehlern. Der 68000 benutzt nur das kurze Stackformat. Ein Teil der 29 Worte sind die Inhalte der Befehls- bzw. Dateneingangspuffer, die Fehleradresse und ein Spezialstatuswort (Bild 4). Es gibt die Art des erfolgten Zugriffs und den Funktionscode an. Durch Ändern des Wortes kann die Programmfortsetzung beeinflußt werden (z. B. Re-Run-Flag = Bit 15 des Wortes wird von der Interruptroutine rückgesetzt; das hat zur Folge, daß der Zugriff nicht wiederholt wird).

Literatur

/1/ Grohmann, B.; Eichler, L.: Das Prozessorbuch zum 68000. Düsseldorf: Data Becker GmbH 1985
/2/ Fäh, P.: Calm – Allgemeine Assemblersprache für Mikroprozessoren. Fribourg 1985
/3/ Fäh, P.: Instruktions Comparsion. Calm-Notation, Fribourg 1985
/4/ Fäh, P.: Description des 68010. Calm-Reference Card, Fribourg 1985
/5/ Rint, C.: Handbuch für den HF- und Elektrotechniker, Band 5. Heidelberg: Hüthig Verlag 1981
/6/ Grafik, W.: Aufbau und Arbeitsweise von 16-Bit-Mikroprozessoren. Berlin: VEB Verlag Technik 1987

Zeitmessung unter C auf dem A 7100

Dr. Burkhard Neumann
Satellitenbodenstation Neustrelitz

Es ist für bestimmte Aufgabenstellungen notwendig, experimentell Zeitdifferenzen zu bestimmen (z. B. Laufzeitmessungen für Datenbank-Operationen). Wird in C programmiert, steht dazu i. allg. die Bibliotheksfunktion CLOCK () zur Verfügung. Dies ist jedoch für den verwendeten Compiler unter SCP auf dem A 7100 nicht der Fall, so daß ein eigenes Bibliotheks-Modul erstellt werden mußte. Dieses sowie der Modus der Parameterübergabe an die aufrufende C-Funktion werden im weiteren erläutert.

Realisierung des CLOCK-Moduls

● Hard- und softwaremäßige Voreinstellungen

Auf der ZVE des A 7100 befinden sich ein programmierbarer Zeitgeber (PIT) 8253, von dessen 3 Kanälen nur der Kanal 2 vom System genutzt wird. Weiterhin ist standardmäßig der Ausgang des PIT-Kanals 0 mit dem Interrupt-Niveau 2 des Interrupt-Controllers (PIC) 8259A verknüpft (X842–X829), so daß für eine interruptgesteuerte Zeitmessung keine Hardware-Modifikationen notwendig sind, wenn der PIT-Kanal 0 als Zeitgeber genutzt wird. Am Takteingang dieses Kanals

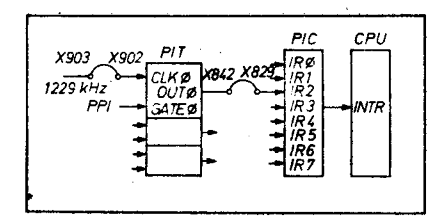


Bild 1 Verknüpfung von PIT und PIC

```
;********   CLOCK-Modul   ********

;CLON:  Initialisierung IV, PIT, PIC; Zaehlerstart
;CLOCK: Zaehlwert ---> AX zur Uebergabe
;CLOFF: Zaehler-Stop; PIT + PIC umprogrammieren

CTOD     EQU     0D0H          ;PIT-Adressen
CTOC     EQU     0D6H

PIC1     EQU     0C0H          ;PIC-Adressen
PIC2     EQU     0C2H

ISRCTO   EQU     88H           ;IV-Position

         TITLE   'CLOCK-MODUL'

         PUBLIC  CLON,CLOCK,CLOFF

         CSEG

CLON:    CLI                   ;Modul-Initialisierung
         PUSH    AX
         PUSH    BX
         PUSH    DS
         MOV     BX,0          ;IV nach IV-Tabelle laden
         MOV     DS,BX
         MOV     DS:ISRCTO[BX],OFFSET ICO
         MOV     DS:ISRCTO+2[BX],SEG ICO
         POP     DS
         POP     BX
         MOV     AL,99H        ;PIC OCW1: IR2 freigeben
         OUT     PIC2,AL
         MOV     AL,35H        ;PIT MW  : Zaehler 0, Mode 2
         OUT     CTOC,AL       ;          4-Dekaden-Zaehler
         MOV     AL,29H
         OUT     CTOD,AL       ;     Teilerfaktor = 1229
         MOV     AL,12H
         OUT     CTOD,AL
         MOV     (ZAEHL),0
         POP     AX
         STI
         RET

CLOCK:   MOV     AX,(ZAEHL)    ;Zaehlwert ---> AX
         RET

CLOFF:   PUSH    AX
         MOV     AL,9DH
         OUT     PIC2,AL       ;PIC: IR 2 sperren
         MOV     AL,30H
         OUT     CTOC,AL       ;PIT stoppen
         POP     AX
         RET
ICO:     PUSH    AX            ;ISR DES CTO
         INC     ZAEHL
         MOV     AL,20H        ;EOI an PIC
         OUT     PIC1,AL
         POP     AX
         STI                   ;INT wieder frei
         IRET

         DSEG

ZAEHL    DW      0             ;Zaehlregister

         END
```

Bild 2 CLOCK-Modul

liegt standardmäßig ein Takt von 1229 kHz an (X903–X902) /1/. Bild 1 veranschaulicht das Zusammenspiel der Komponenten.

● Programmierung von PIT und PIC

Der Start des CLOCK-Moduls (Bild 2) erfolgt durch Aufruf von CLON. Als erstes ist in der Interrupt-Vektortabelle für den Interrupt-Code 22H (≙ System-PIC-Initialisierung) die Startadresse der Service-Routine (IP und CS) abzulegen. Weiterhin ist eine Ummaskierung des PIC-Steuerwortes OCW1 so vorzunehmen, daß IR2 freigegeben wird sowie die anderen, vom System benötigten Interrupt-Niveaus erhalten bleiben. Anschließend kann PIT0 als Zeitgeber im Mode 2 programmiert werden; der Teilerfaktor ist entsprechend dem angestrebten Zeitintervall zu wählen. Der genaue Ablauf ist anhand des Beispielprogramms in Bild 3 ersichtlich. Hier wurde der Teilerfaktor so gewählt, daß ein Interrupt-Intervall von 1 ms erzeugt wird.
Die Übergabe der Zeitintervallanzahl erfolgt bei Aufruf der Funktion CLOCK.
Wichtig ist, daß vor Beendigung des das CLOCK-Modul aufrufenden Programms das Interrupt-Niveau 2 des PIC wieder gesperrt wird, da es ansonsten beim Laden weiterer Programme zu System-Fehlreaktionen kommt. Diese Aufgabe erfüllt die Funktion CLOFF.

```
/* C-Testprogramm fuer Clock-Modul
   *******************************

   CLOCK-Zeitraster: 1 ms

   CLON:  Initialsierung des CLOCK-Moduls
   CLOCK: Parameteruebernahme
   CLOFF: Abschalten des CLOCK-Moduls
*/

#include <stdio.h>

WORD    a,c,CLON(),CLOCK(),CLOFF();

main()
{
   printf("\n\nC-Testprogramm fuer Clock-Modul");
   printf("\n\nStart durch Tastendruck");
   getchar();
   CLON();
   a=c=0;
   while (a < 30000)
   {
      a=CLOCK();
      if(c < a/1000)
      {
         c+=1;
         printf("\nDelta t (in s) = %4d",c);
      }
   }
   CLOFF();
}
```

Bild 3 C-Testprogramm

Tafel 1 Nutzung der 8086-Register zur Parameterübergabe

[illegible]	[illegible]
(unsigned) (short)int	*AX*
long int	*BX, AX*
float	*BX, AX*
double	*DX, CX, BX, AX*

Parameterübergabe

In C übergeben bzw. übernehmen Funktionen bekanntermaßen keine Parameter-Adressen, sondern Werte („call by value“); dieses kann über den Stack oder über Register erfolgen. Im vorliegenden Fall ist nur die Intervall-Anzahl seit dem Zeitgeberstart von der Funktion CLOCK zu übergeben; die Übergabe erfolgt im Beispielprogramm für einen Integer-Wert im Register AX. Für weitere Datentypen ist die Registernutzung in Tafel 1 angegeben.
Für das problemlose Linken der C- und Assembler-Module seien noch folgende Hinweise gegeben:
– Der Funktionsname der Assembler-Module muß im Assemblerprogramm als PUBLIC deklariert werden.
– Da verschiedene Compiler unterschiedlich auf Groß- und Kleinschreibung reagieren, sollte man die Funktionsnamen immer groß schreiben.

Literatur

/1/ Betriebsdokumentation AC A 7100 Band 1. VEB Robotron-Elektronik Dresden, 1986

✉ KONTAKT ☎

Institut für Kosmosforschung der AdW, Satellitenbodenstation, Kalkhorstweg 53, Neustrelitz, 2080; Tel. 7481

Wegbereiter der Informatik



PIERRE SIMON MARQUIS de LAPLACE

* 1749 Beaumont-en-Auge (Normandie),
† 1827 Paris

Die großen Naturforscher des 17./18. Jahrhunderts besaßen über ihre Fachdisziplinen hinaus zumeist eine universelle Bildung, die es ihnen erlaubte, auf verschiedensten Gebieten wissenschaftliche Leistungen zu vollbringen. Auch der französische Mathematiker, Physiker und Astronom P. S. Laplace hat eine vielseitige Ausbildung erhalten. Er besuchte ab 1755 eine Schule des Benediktinerordens und trat 1765 in das Jesuiten-Kolleg zu Cáen ein, wo er alte Sprachen, Literatur, Kunst, Mathematik und Astronomie studierte; 1768 setzte er seine Studien in Paris fort. Auf Vermittlung von d'Alembert, dem ständigen Sekretär der Pariser Akademie und dem bis dahin wohl einflußreichsten Mathematiker des damaligen Frankreich, erhielt Laplace 1772 ein Lehramt an der Pariser Militärakademie, wo 1784/85 Napoleon zu seinen Schülern zählte. An der 1794 neu gegründeten technischen Hochschule in Paris (Ecole Polytechnique) wurde er Professor für Mathematik. Als Vorsitzender der Kommission für Maße und Gewichte hatte er wesentlichen Anteil an der Einführung eines einheitlichen dezimalen Maßsystems. Im Jahre 1799 ernannte ihn Napoleon zum Innenminister und berief ihn kurz danach in den Senat. Mit 24 Jahren wurde er bezahltes Mitglied der Akademie; auch war er Ehrenmitglied der Akademie in Petersburg.
Laplace hat besonders in der Mathematik und in der mathematischen Physik wesentliche Beiträge geleistet. Er konstruierte zwar keine Rechenmaschine, doch wird ihm eine andere bedeutende Erfindung zugeschrieben, die sich ebenfalls als ein hervorragendes Rechenhilfsmittel herausstellte und sich bis heute bei Aufgaben der elektrischen Netzwerke, der Regelungstechnik und bei der Simulation technischer Vorgänge als außerordentlich brauchbar erwiesen hat. Diese Erfindung betrifft die zumindest jedem Automatisierungstechniker geläufige Transformation

$$X(p) = \int_0^\infty e^{-pt}\, x(t)\, dt,$$

welche einer Funktion x(t) der reellen Variablen t die Funktion X(p) der komplexen Variablen p zuordnet (vorausgesetzt, das uneigentliche Integral existiert). Der wissenschaftliche Background, der mit dieser Formel verbunden ist, wird als *Theorie der Laplace-Transformation* gegenwärtig an den Hochschulen für Mathematiker, Elektronikingenieure und Regelungstechniker gelehrt. Und das mit gutem Grund! Gestattet doch diese Transformation, bestimmte Problemklassen gewöhnlicher Differentialgleichungen lediglich mit rein algebraischen Rechenoperationen auflösen zu können, was eine bedeutende Reduzierung des Rechenaufwandes mit sich bringt. Es sei erwähnt, daß die Auflösung von Differentialgleichungen mit dieser Laplaceschen Methode (wenn sie anwendbar ist) sogar stets in exakter Weise gelingt – während sie mit computertechnischen Hilfsmitteln ja prinzipiell immer nur näherungsweise möglich ist.
Eine weitere wichtige Anwendung erfährt obige Transformation noch heute in der Automatisierungstechnik: mit ihrer Hilfe wird die Übertragungsfunktion eines dynamischen Systems definiert; und sie ermöglicht z. B. auch den Beweis für die Tatsache, daß ein unbekanntes lineares Übertragungssystem mit der Systemantwort auf ein einziges Eingangssignal sein gesamtes Übertragungsverhalten „verrät“. Übrigens ist mit den heute zur Verfügung stehenden computertechnischen Möglichkeiten sogar eine Automatisierung des gesamten Laplace-Kalküls in greifbare Nähe gerückt.
Laplace verfaßte sehr zahlreiche Arbeiten über Mathematik und experimentelle und mathematische Physik (14 Bände) sowie über Himmelsmechanik (5 Bände). So untersuchte er partielle Differentialgleichungen 2. Ordnung auf ihre Lösungsmöglichkeit, ersann die Kaskadenmethode – ein Lösungsverfahren für hyperbolische Differentialgleichungen – und befaßte sich mit partiellen Differenzengleichungen. Zur Lösung der Laplaceschen Differentialgleichung lassen sich Analogie-Netzwerke konstruieren, die bei ADLER („Analogrechnen“) beschrieben werden. Laplace entwickelte auch eine Kapillartheorie für Flüssigkeiten, leitete eine Formel für die Ausbreitungsgeschwindigkeit des Schalls in der Luft ab und verbesserte die barometrische Höhenformel. Desweiteren erforschte er die Umlaufzeiten von Jupiter, Saturn und den drei inneren Jupitermonden und machte sich Gedanken über die Stabilität des Sonnensystems.
Bei seinem Begräbnis erwies ihm übrigens kein Geringerer als Alexander von Humboldt die letzte Ehre, der zwecks Herausgabe seiner eigenen Werke mehrere Jahre in Paris weilte.

Dr. Klaus Biener

Mikroprozessorsystem K 1810 WM 86

Hardware · Software · Applikation (Teil 4)

Prof. Dr. Bernd-Georg Münzer
(wissenschaftliche Leitung),
Dr. Günter Jorke, Eckhard Engemann,
Wolfgang Kabatzke, Frank Kamrad,
Helfried Schumacher, Tomasz Stachowiak
Wilhelm-Pieck-Universität Rostock,
Sektion Technische Elektronik,
Wissenschaftsbereich Mikrorechentechnik/
Schaltungstechnik

Für den Fall, daß die Datei eine neu zu erstellende Datei ist, wird folgende Mitteilung mit ausgegeben:



ED erlaubt eine Reihe von Kommandos. Die *wichtigsten* sollen nachfolgend erläutert werden.

6.2.1 Texttransferkommandos

Die möglichen Kommandos sind:

nA *Append*
Anfügen von n unbearbeiteten Quellzeilen aus der Quelldatei an den gefüllten Teil des Arbeitspuffers. Wird für nA ein **#A** eingegeben, werden zirka 48 KByte als Textpuffer reserviert. Kleine Dateien können mit #A zum Beginn des Editiervorganges komplett geladen werden. Wird für nA ein **0A** eingegeben, werden nur so viel Textzeilen eingelesen, bis der Arbeitspuffer halb gefüllt ist. Wird kein n eingegeben, liest ED nur eine Quellzeile ein.

nW *Write*
Schreiben der ersten n Zeilen zur temporären Hilfsdatei. Diese Zeilen werden in der neuen Datei an das Ende der schon eingetragenen Zeilen angefügt. Wird für nW ein **0W** eingegeben, werden soviel Zeilen in die neue Datei geschrieben, bis der Arbeitspuffer etwa halb leer ist. Nach der Ausführung eines W-Kommandos muß das **H**-Kommando gegeben werden, wenn erneut auf die geretteten Zeilen zugegriffen werden soll.

E *Exit*
Beenden von ED. Es werden die gesamten gepufferten Zeilen und die unbearbeiteten Zeilen zur neuen Datei kopiert. Dabei wird die alte Datei in eine Datei mit dem Dateisuffix .BAK umgewandelt.

6.2.2 Editiergrundkommandos

B *Begin*
–B
Bewegen des Characterpointers (**CP**) an den Beginn (**B**) oder das Ende (**–B**) des Arbeitspuffers.

nC
–nC
Bewegen des **CP** um n Zeichen vorwärts oder rückwärts.

nD *Delete*
–nD
Löschen von n Zeichen vor (**nD**) oder hinter (**–nD**) dem CP.

nK *Kill*
–nK
Löschen von n Zeilen vor dem **CP** (**nK**) oder hinter dem **CP** (**–nK**).

nT *Type*
–nT
Anzeige von n Zeilen vor dem **CP** (**–nT**) oder hinter dem **CP** (**nT**) auf dem Terminal.

n
–n
Bewegen des CP um n Zeilen vor den **CP** (**–n**) oder hinter den **CP** (**n**). Ein Betätigen von **CR** (**ENTER**) wird wie die Eingabe des Kommandos

:1

interpretiert, d. h., der **CP** wird jeweils um eine Zeile weiter gestellt.

I *Input*
i
Einstellen des Eingabemodus. Es werden solange Textzeilen **vor** den **CP** eingegeben, bis die Eingabe von **CTRL–Z** erfolgt ist.

6.2.3 Kommandos zur Modifizierung des Arbeitspuffers

S *Substitute*
s
Ersetzen von Zeichen und Zeichenketten im Arbeitspuffer. Das Kommando wird wie folgt gegeben:
nssearch_string^znew_string
wobei n die Anzahl der Substitutionen angibt.



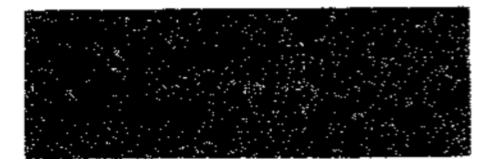
Es erfolgt keine Anzeige der geänderten Zeile. Soll diese mit angezeigt werden, muß eine Kommandoverkettung in folgender Weise eingegeben werden:



Mit Hilfe des S-Kommandos lassen sich auch Zeichen und Zeichenketten streichen.



Weiterhin ist eine Kommandoverkettung auf der Basis der **C-**, **D-**, **L-**, **K-** und **T**-Kommandos möglich.





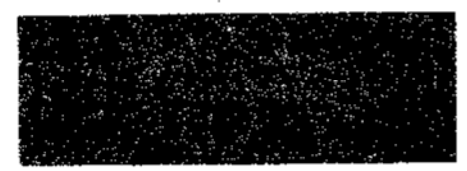
6.2.4 Fortgeschrittene ED-Kommandos

nP Page
–nP
Dieses Kommando bewirkt eine komplette bildschirmfüllende Anzeige und bewegt den CP an den Anfang einer neuen Bildschirmseite. Wenn keine Seitenzahl angegeben wurde, wird der CP um 23 Zeilen vorwärts bewegt und die weiteren 23 folgenden Zeilen werden angezeigt. Mit 0P kann die aktuelle Seite angezeigt werden, ohne daß der CP weiter bewegt wird.

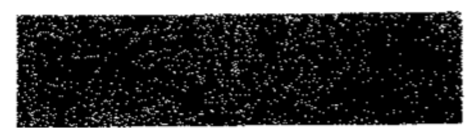
n: Line Number
Mit Hilfe dieses Kommandos kann der CP auf eine spezifizierte Zeile eingestellt werden.



:n Trogh Line Number
Das zum Line-Number-Kommando inverse Kommando wird bis zu einer bestimmten Zeilennummer ausgeführt.



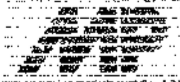
4: *
nFstring{^z}
Suchfunktion, wobei n eine Zahl ist, die das n-te Auftreten der gesuchten Zeichenkette angibt. Wichtig hierbei ist, daß die Zahl n eine positive Zahl ist, da **ED** im Puffer nur *vorwärts* suchen kann.



Der CP wird auf die Zeile *nach* der gesuchten Zeichenkette positioniert, wo (in diesem Beispiel) das dritte Auftreten der Zeichenkette zyklisch lokalisiert wurde. Durch die Kombination des **F**- mit dem **T**-Kommando kann die Zeile mit angezeigt werden:



Wenn ED die Zeichenkette im Arbeitspuffer



nicht findet, wird eine Fehlermeldung ausgegeben (siehe 6.2.7).

nNstring{^z}

Das N-Kommando sucht über den Arbeitspuffer hinaus in der *gesamten* Quelle. Der Zeichenkette muß ein CTRL-Z folgen, wenn noch ein weiteres Kommando angehängt werden soll. Wenn das Editieren fortgesetzt werden soll, nachdem die Quelldatei bearbeitet und der Arbeitspuffer geleert ist, **muß** das H-Kommando angewendet werden.

6.2.5 Transport von Textblöcken

Diese Kommandos werden benutzt, um eine Anzahl von Textzeilen von einem Dateibereich in einen anderen zu transportieren. Hierzu muß zuerst das X-Kommando benutzt werden, welches den gewünschten Textblock in eine Temporärdatei (XOOOOOOO.LIB) schreibt. Danach werden die Originalzeilen mit dem K-Kommando aus dem Text gelöscht und der Textblock mit dem R-Kommando an das gewünschte Ziel geladen.

nX

nXfilespec

Die Zahl n gibt die Anzahl der Zeilen ab CP in Richtung Pufferende an, die in die Temporärdatei zu übertragen sind (n **muß** positiv sein). Wird kein Dateiname angegeben, nimmt ED automatisch XOOOOOOO.LIB an. Ist kein Dateisuffix spezifiziert, wird automatisch .LIB zugewiesen.

R

Rfilespec

Mit dem R-Kommando können die Zeilen zurück übertragen werden, die mit dem **X**-Kommando in die Temporärdatei geschrieben wurden bzw. es können andere Quelltextdateien in die zu bearbeitende eingelesen werden. Ein nicht angegebenes Dateisuffix wird durch **.LIB** ersetzt. Das **R**-Kommando liest die Datei **vor** den **CP** ein.

6.2.6 Beendigung von ED und Dateisicherung

H *Head of File*

Ein H-Kommando sichert den Inhalt des Arbeitspuffers, ohne jedoch den Editiervorgang zu beenden und kehrt zum Kopf (Head) der Datei zurück. Auf die bereits editierten Daten kann erneut zurückgegriffen werden.

O *Original*

Ein O-Kommando schließt den Editiervorgang ab und läßt die Datei im Originalzustand zum erneuten Editieren, ohne **ED** abzubrechen. ED beantwortet die Eingabe des O-Kommandos mit der Frage:

O(Y/N)?

Diese muß mit Y beantwortet werden, wenn das O-Kommando ausgeführt werden soll. Mit N kehrt man in den normalen ED-Betrieb zurück. Jede andere Taste wiederholt die Frage.

Q *Quit*

Ein Q-Kommando schließt den Editiervorgang ab und beendet ED. Bei der Eingabe des Q-Kommandos wird ebenfalls abgefragt:

Q(Y/N)?

Geantwortet werden muß mit **Y** oder **N**, jedes andere Zeichen wiederholt die Frage. Die Temporärdatei wird gelöscht, die Quelldatei wird geschlossen, und es wird keine Backup-Datei mit dem Dateisuffix .BAK erzeugt.

E *Exit*

(siehe 6.2.1)

6.2.7 ED-Fehlermeldungen

Die allgemeine Form der **ED**-Fehlermeldung ist:

BREAK "x" **AT** c,

wobei x ein Fehlersymbol darstellt und c der Buchstabe des Kommandos ist, bei dem der Fehler aufgetreten ist.

Fehlersymbol	Bedeutung
#	Suchfehler
?c	unbekannter Kommandobuchstabe
O	keine **.LIB**-Datei
>	Puffer voll
E	Kommandoabbruch. Eine Tastatureingabe hat das Kommando abgebrochen
F	Dateifehler. Es erfolgt entweder DISK FULL oder **DIRECTORY FULL.**

Hinweis:

ED interpretiert die Kommandoeingaben je nachdem, ob sie als Groß- oder Kleinbuchstaben erfolgten. Es ist die generelle Eingabe als Kleinbuchstaben zu empfehlen, da hierbei **alle** Buchstaben, ob Klein- oder Großbuchstaben im Text gesucht, gefunden oder bearbeitet werden können. Eine Kommandoeingabe als Großbuchstabe erlaubt nur die Bearbeitung von Texten, die generell aus Großbuchstaben und Ziffern bestehen.

6.3 Der Absolutassembler ASM86 und der Relativassembler RASM86

ASM86 und **RASM86** wurden für die Realisierung verschiedener Aufgaben geschaffen:

ASM86

Assemblierung von absoluten Programmen. Eine modulare Programmierung ist nicht oder nur sehr schlecht möglich.

RASM86

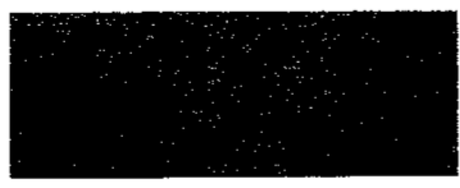
Assemblierung von verschieblichen Programmen und Programmodulen. Eine Anbindung an Programme, die von anderen Übersetzern erzeugt wurden (z.B. **PASCAL, C, FORTRAN77**) ist möglich.

6.3.1 Bedienung

Die Assembler **ASM86** und **RASM86** werden nach einem einheitlichen Konzept bedient. Die Kommandozeile sieht allgemein wie folgt aus:



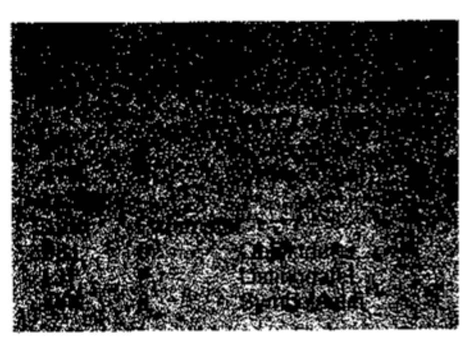
Es gibt bei den Dateinamen eine Standardannahme in der Weise, daß die Quelldateien mit dem Dateisuffix .A86 versehen sind. In diesem Falle braucht das Dateisuffix beim Assembleraufruf in der Angabe **name** nicht mit angegeben werden. Sollen Dateien mit einem anderen Suffix assembliert werden, so ist dieses mit anzugeben. Nach ihrem Aufruf melden sich die Assembler mit einer Identifikationsausschrift:



wobei die Angabe x.x die jeweils eingesetzte Versionsnummer angibt. Stimmt der Dateiname nicht oder ist er nicht auf dem angegebenen Laufwerk vorhanden, geben beide Assembler die Fehlermeldung

NO FILE

aus und brechen die Arbeit ab. Entsprechend dem jeweiligen Assemblerkonzept können die Assembler aus dem Quelltext verschiedene Ergebnisdateien erzeugen:



Die Angabe der Optionen kann unter Angabe der oben angeführten Parameter und zusätzlicher Geräteparameter die Erzeugung und Abspeicherung der zu erzeugenden Dateien beeinflussen. Als Geräteparameter können die Buchstaben

A, B, . . . P oder **X, Y, Z**

angegeben werden, wobei

A bis P die Laufwerke spezifizieren

X die Bedienkonsole **(CON:)**

Y den Drucker **(LST:)**

Z die Ausgabe unterdrückt **(NUL:)**.

Beim RASM86 ist zusätzlich noch die Option "L" möglich, mit deren Hilfe die lokalen Parameter in die Objektdatei eingeschlossen werden können, welche in der durch **LINK86** erzeugten Symboldatei erscheinen. Andernfalls erscheinen nur **PUBLIC**-Symbole in der Symboldatei. Die erzeugten SYM-Dateien sind für die Arbeit mit dem symbolischen Debugger **SID86** (siehe Abschnitt 6.5) notwendig. Bei der Ausgabeumlenkung auf die Konsole kann die Ausgabe mit **CNTRL-S** angehalten werden und mit **CNTRL-Q** fortgesetzt werden.



dem Laufwerk **C** und erzeugen von TEST.H86 und **TEST.LST. TEST.SYM** wird unterdrückt.

ASM86 TEST ○PY SX

Assemblieren der Datei **TEST.A86** und erzeugen von **TEST.H86.** Die Ausgabe von **TEST.LST** erfolgt direkt auf den Drucker und die Ausgabe von **TEST.SYM** erfolgt direkt auf die Konsole.

RASM86

RASM86 TEST

Assemblieren der Datei **TEST.A86** und erzeugen von **TEST.OBJ, TEST.LST** und **TEST.SYM** auf dem aktuellen Laufwerk.

RASM86 TEST ○AC SZ PZ

Assemblieren von **TEST.A86** auf dem Laufwerk **C** und erzeugen von **TEST.OBJ.** Die Dateien **TEST.SYM** und **TEST.LST** werden unterdrückt.

RASM TEST ○LO

Assemblieren von **TEST.A86** auf dem aktuellen Laufwerk und erzeugen von **TEST.OBJ, TEST.LST** und **TEST.SYM.** Die lokalen Symbole werden in **TEST.OBJ** mit eingeschlossen.

ASM86 und RASM86 können durch Betätigung einer beliebigen Taste angehalten werden. Beide Assembler antworten auf eine Tastenbetätigung mit:



Ein Y bricht den Assemblerlauf ab. Ein N setzt die Arbeit fort.

6.3.2 Elemente der Assemblersprachen für ASM86 und RASM86

Beide Assembler stellen hinsichtlich der Assemblernotation eine Einheit dar. Es gibt keine Unterschiede. Da eine ausführliche Beschreibung am Beispiel des ASM86 in Kapitel 5 erfolgt ist, soll in diesem Kapitel nicht näher darauf eingegangen werden.

6.3.3 Assemblerdirektiven

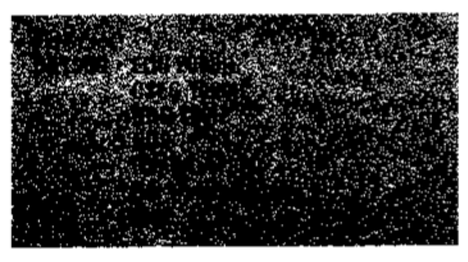
Direktivanweisungen sind wichtig für die Zuweisung von Codeteilen zu logischen Segmenten. Weiterhin lassen sich mit solchen Anweisungen die bedingte Assemblierung, die Datenelementdefinition und die Listenformatsteuerung durchführen. RASM86 enthält im wesentlichen den Direktivanweisungssatz des ASM86, so daß hier nur die wesentlichen Unterschiede bzw. Ergänzungen aufgezeigt werden sollen. Nachfolgend wird eine Übersicht der Direktivanweisungen beider Assembler gegeben.

	RASM86	**ASM86**
CSEG	x	x
DSEG	x	x
SSEG	x	x
ESEG	x	x
ORG	x	x
END	x	x
IF	x	x
ENDIF	x	x
ELSE	x	–
EQU	x	x
DB	x	x
DW	x	x
DD	x	x
RS	x	x
RB	x	x
RW	x	x
RD	x	–
EJECT	x	x
LIST	x	x
NOLIST	x	x
IFLIST	x	–
NOIFLIST	x	–
PAGESIZE	x	x
PAGEWIDTH	x	x
TITLE	x	x
INCLUDE	x	x
SIMFORM	x	x
NAME	x	–
PUBLIC	x	–
EXTRN	x	–
GROUP	x	–

Die geringfügigen Unterschiede bei der Anwendung der Segmentdirektiven und der erweiterte Direktivsatz des RASM86 sollen nachfolgend erläutert werden.

● **Segmentdirektiven**

Die beiden Assembler benötigen die Segmentdirektiven zur Unterscheidung der Code-, Daten-, Stack- und Extrasegmentbereiche. Beim ASM86 können in den Direktivanweisungen numerische Ausdrücke bzw. Operatoren zur Festlegung der Segmentposition angegeben werden.



Die Segmentdirektiven des **RASM86** sind daher dieser Aufgabenstellung mit angepaßt. Segmente können nach folgendem Schema benannt werden:

[seg_name] segment [align_typ]
[combine_typ][class_name]

Als **segment** ist möglich: **CSEG**
DSEG
SSEG
ESEG.

Der Segmentname **seg_name** kann ein beliebiger RASM86-Bezeichner sein, wobei bei Nichtangabe folgende Standardnamen von RASM86 angenommen werden:

Segment-Direktive	Standardnamen
CSEG	**CODE**
DSEG	**DATA**
SSEG	**STACK**
ESEG	**EXTRA**

RASM86 verbindet alle Segmente mit dem gleichen Segmentnamen, auch wenn sie nicht zusammenhängend im Quellcode stehen. Für den Linker **LINK86** kann durch die Angabe des Zuordnungstypes **(align_typ)** eine spezielle Segmentadresse festgelegt werden. Folgende Angaben sind möglich:

BYTE Bytezuweisung
(Segment beginnt ab dem nächsten Byte)

WORD Wortzuweisung
(Segment beginnt an einer geraden Adresse)

PARA Paragraphenzuweisung
(Segment beginnt an einer Paragraphenadresse [A0 – A3 = 0])

PAGE Seitenzuweisung
(Segment beginnt an einer Seitengrenze [A0 – A7 = 0])

Wird kein Typ explizit angegeben, wird folgende Standardtypenzuordnung wirksam:

Segment-Direktive	Zuordnungstyp
CSEG	**BYTE**
DSEG	**WORD**
SSEG	**WORD**
ESEG	**WORD**

Die Angabe des Verbindungstypes **(combine_typ)** ist für den Linker wichtig, damit er Segmente mit anderen Segmenten gleichen Namens verbinden kann. Folgende Verbindungstypen sind angebbar:

PUBLIC
Alle Segmente dieses Verbindungstypes werden in der Reihenfolge ihres Auftretens durch den Linker verkettet. (Standardtyp, wenn keine Angabe erfolgt).

COMMON
Das Segment hat gleiche Speicherplätze mit anderen Segmenten gleichen Namens

STACK
Alle Segmente eines Verbindungstypes werden zu höheren Adressen hin überlagert, weil Stacks abwärts zu niederen Adressen hin wachsen.

LOCAL
Das Segment ist lokal im übersetzten Programm. Es wird nicht mit anderen Segmenten verbunden.

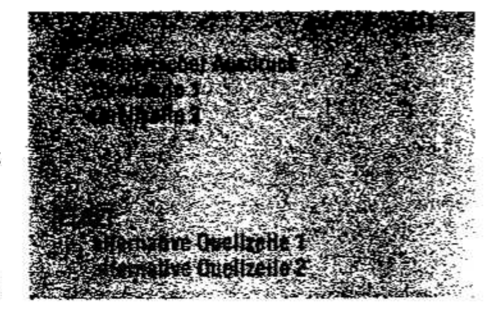
nnnn
Für absolute Programme bestimmt RASM86 die Ladeadresse während des Assemblerlaufes. Die Position wird durch den Linker zum Ladezeitpunkt bestimmt.

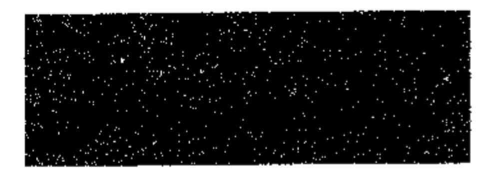
Durch die Angabe des Klassennamens **class_name** können gekennzeichnete Segmente im gleichen Bereich einer vom Linker erzeugten CMD-Datei eingeordnet werden. Falls nicht durch die GROUP-Direktive oder durch Linkerkommandos eine Umstellung erfolgt, ordnet RASM86 die Segmente in die CMD-Datei wie folgt ein:

Segment-Direktive	Standard-Klassenname	CMD-Bereich
CSEG	**CODE**	**CODE**
DSEG	**DATA**	**DATA**
SSEG	**STACK**	**STACK**
ESEG	**EXTRA**	**EXTRA.**

● **ELSE-Direktive**

Im Gegensatz zum ASM86 kann eine alternativ zu übersetzende Folge von Quelltextzeilen angegeben werden, welche übersetzt wird, falls die angegebene Bedingung unwahr ist. Es ergibt sich eine einfachere Bedienung als beim ASM 86.





Die nicht übersetzten Quellzeilen werden aber mit in die Listingdatei aufgenommen, d. h. nur gedruckt, sofern diese nicht unterdrückt werden.

● **RD-Direktive**

[symbol] RD numerischer Ausdruck

Die RD-Direktive reserviert ein Doppelwort im Speicher ohne Initialisierung.

● **IFLIST-/NOIFLIST-Direktive**

Die NOIFLIST-Direktive unterdrückt die Ausgabe des während der bedingten Assemblierung zu übersetzenden Blockes, wenn *keine* Übersetzung angewiesen wurde. Die Ausgabe kann durch die IFLIST-Direktive angewiesen werden.

● **GROUP-Direktive**

group_name **GROUP** seg_name **1**, . . . ,seg_name n

Die **GROUP**-Direktive weist **RASM86** an, die aufgeführten Segmente in eine Gruppe zu verbinden. Die Reihenfolge der Namen bestimmt, in welcher Reihenfolge der Linker die Segmente in die CMD-Datei legt.

● **NAME-Direktive**

NAME 'modul_name'

Die **NAME**-Direktive weist dem von RASM86 erzeugten Objektmodul einen Namen zu. Sofern keine Angabe erfolgt, wird der Name der Quelldatei als Name des Objektmoduls übernommen.

● **PUBLIC-Direktive**

PUBLIC name,{name, . . .}

Die **PUBLIC**-Direktive legt fest, daß mit **PUBLIC** definierte Namen zu anderen miteinander verbundenen Programmen Bezug nehmen können.

● **EXTRN-Direktive**

EXTRN external_id[,external_id, . . .]

Diese Direktive ermöglicht RASM86 den Zugriff innerhalb des übersetzten Programmes zu *jedem* externen Symbol, welches in einem anderen Programm definiert ist. Der external_id besteht aus einem Symbol und einem Typ. Das Symbol kann eine Zahl oder Variable sein. Als Typ sind möglich:



6.4 Das Filegenerierprogramm GENCMD und der Linker LINK86

6.4.1 GENCMD

Mit Hilfe des Programmes GENCMD können Dateien im H86-Format (entstanden z. B. in einem Assemblerlauf mit **ASM86**) in CMD-Dateien, die auf dem Betriebssystem SCP **1700** abarbeitungsfähig sind, umgesetzt werden. **H86**-Dateien können auch vor der Bearbeitung durch **GENCMD** mit Hilfe des Programmes **PIP** zu größeren **H86**-Dateien verkettet werden. Es muß hierbei beachtet werden, daß die einzelnen **H86**-Dateien, die zusammengefaßt werden sollen, Speicherbereiche überlagern können. Bei einer Speicherinhaltsdefinition in einer **H86**-Datei ist für **GENCMD** die *letzte* Definition gültig. Diese Eigenschaft wird beispielsweise bei der Implementation des Betriebssystemes **SCP 1700** ausgenutzt. **GENCMD** wird nach folgendem Schema aufgerufen:

GENCMD datei parameterliste

wobei **datei** der Name einer **H86**-Datei ist (Einzel- oder Überlagerungsdatei) und **parameterliste** Schlüsselwörter angibt, die durch Komma oder Leerzeichen getrennt werden. Mögliche Schlüsselwörter sind:

8080 bewirkt die Bildung einer **CMD**-Datei in Form eines **8080**-Speichermodells, bei dem Code- und Datenbereich innerhalb eines Segmentes vermischt sind, unabhängig davon, ob im Quellprogramm **CSEG**- und **DSEG**-Anweisungen verwendet sind.

CODE	Codesegmentzuweisung
DATA	Datensegmentzuweisung
STACK	Stacksegmentzuweisung
EXTRA	Extrasegmentzuweisung
X1, X2, X3, X4	Hilfsbereichzuweisung

Den Schlüsselwörtern, die den jeweiligen Bereich definieren, folgt eine Hexadezimalzahl, die eine Paragraphenadresse oder Segmentlänge in Paragraphen angibt. Diese Werte sind in eckige Klammern einzuschließen und werden durch Komma getrennt. Diesen Werten wird ein einzelner Buchstabe vorangestellt, welcher die Bedeutung des jeweiligen Wortes definiert:

Axxxx Laden auf Absolutadresse xxxx

Bxxxx Bereich beginnt auf xxxx in der **H86-Datei**

Mxxxx Bereich erfordert Speichergröße von minimal xxxx * 16 Byte (xxxx * Paragraphen)

Xxxxx Bereich erfordert Speichergröße von maximal xxxx * 16 Byte (xxxx * Paragraphen)

Hinweis:

– Der A-Wert *muß* für jeden Bereich angegeben werden, der im Speicher fest lokalisiert werden soll. Im *Normalfall* braucht man diesen Wert nicht anzugeben, da SCP 1700 den TPA-Raum intern verwaltet und den Programmen den Speicher zuweist

– Ein B-Wert *muß* angegeben werden, wenn **GENCMD** Dateien verarbeitet, die nicht mit **ASM86** erzeugt sind, z. B. bei Dateien, die keine Informationen zur automatischen Segmentbereichsunterscheidung enthalten.

Beispiele:

B>A:GENCMD SCP 8080 CODE[A40]

Die Datei **SCP.H86** wird in die Datei **SCP.CMD** umgewandelt, wobei das **8080**-Speichermodell zugrunde gelegt wird. Der Codebereich beginnt auf der Paragraphenadresse 40H.

B>A:GENCMD C:TEST CODE[A1000] DATA[M20] EXTRA[B40] STACK[M80] X1[M80]

Eine auf dem Diskettenlaufwerk C befindliche Datei **TEST.H86** wird in die Datei **TEST.CMD** umgewandelt. Der Codebereich beginnt bei der Paragraphenadresse **1000H**, der Datenbereich erfordert einen Speicherbedarf von mindestens **20H** Paragraphen. Der Extrabereich beginnt ab der Paragraphenadresse 40H in der H86-Datei und der Stack- sowie der Zusatzbereich erfordern eine Mindestgröße von **80H** Paragraphen.

6.4.2 LINK86

● **Bedienung**

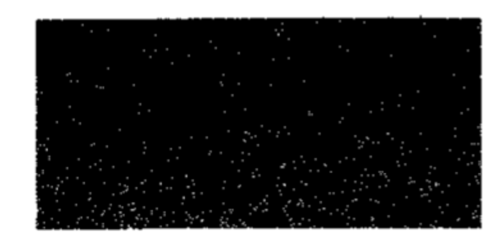
LINK86 ist ein Dateiverbindungsprogramm, welches verschiedene Objektmodule zu einer **CMD**-Datei verbindet. Hierbei ist es gleichgültig, ob die Objektmodule mit Hilfe des **RASM86** oder anderen Übersetzern (z. B. **FORTRAN 77, C, PASCAL**) erzeugt wurden. Weiterhin erlaubt **LINK86** das Einbinden von Bibliotheksmodulen (**L86**-Dateien), welche indiziert aufgebaut sind. **LINK86** erzeugt drei Dateitypen:

Kommandodatei	(**CMD**-Datei)
Symboltabellendatei	(**SYM**-Datei)
Listdatei	(**MAP**-Datei).

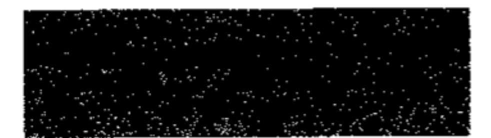
LINK86 wird nach folgendem Bedienschema aufgerufen:

LINK86 {file=}file1{,file2, . . . ,filen}

Wird ein Name vor dem Gleichheitszeichen angegeben, erzeugt **LINK86** die geforderten Ausgabedateien mit ihren dazugehörigen Dateitypen. Wird kein neuer Name angegeben, werden die von **LINK86** erzeugten Dateien mit dem ersten Namen aus der Kommandozeile versehen.



Weiterhin erlaubt **LINK86** das Auslesen öfter wiederkehrender Kommandozeilen aus einer Eingabedatei. Derartige Dateien müssen den Dateityp **INP** haben.



LINK86 kann während seiner Arbeit durch Betätigen einer beliebigen Taste angehalten werden. **LINK86** meldet sich mit folgender Ausschrift:



Die Eingabe von Y veranlaßt **LINK86**, die Arbeit abzubrechen und zum **SCP 1700** zurückzukehren. Ein N setzt die Arbeit von **LINK86** fort.

wird fortgesetzt

Werkzeuge zur hardwarenahen Programmierung in höheren Programmiersprachen auf 8-Bit-Computern

Hardwarenahe und betriebssystemnahe Programmierung ist traditionsgemäß der Bereich der **Systemprogrammierung,** *der wesentlich auf dem Einsatz der Assemblersprachen basiert. In der Entwicklungsgeschichte höherer Programmiersprachen kann man eine Tendenz zur Unterstützung hardwarenaher Programmierung (hrp::= hardware related programming) verfolgen, die sich heute auf 8-Bit-Computern u. a. in BASIC und TURBO-PASCAL als soweit praxisrelevant manifestiert, daß z. B. auf dem Gebiet der* **Anwender-Systemprogrammierung** *eine Kombination aus höheren Sprachen und unterstützenden Assemblerroutinen gegenüber der reinen Assemblerprogrammierung wesentliche Vorteile bietet.*

In einer Folge mit den Beiträgen

1. „Umwandeln von (Assembler-)COM-Files in dBASE-II-INLINE-POKEs“

2. „EXEC – ein Startprogramm für dBASE II“

3. „Automatische Generierung von BASIC-INLINE-POKEs“

4. „Externe (Assembler-)Unterprogramme in TURBO-PASCAL“

werden in mehreren Heften der MP unmittelbar nutzbare Werkzeuge in Form von Programmen unter CP/M-kompatiblen Betriebssystemen für 8-Bit-Computer vorgestellt.

Neben den **Programmen** *werden methodische und programmiertechnische Erfahrungen dargestellt. Als höhere Programmiersprachen sind dBASE II, BASIC-80 und TURBO-PASCAL ausgewählt worden, für die INLINE- und externe Unterprogramme auf dem Niveau der höheren Sprache sowie deren Einbindung aufgezeigt werden.*

Vom Autor kann bei Bedarf eine Demo-Diskette mit dem Inhalt dieser Artikelserie bespielt werden.

Umwandeln von COM-Files in dBASE-II-INLINE-POKEs

Christian Hanisch, Berlin

Der Einsatz von Assemblerroutinen z. B. ab der Adresse 0A400H bringt bei dBASE-Programmen oft erhebliche Effektivitätsverbesserungen bzw. erhöhte Nutzerfreundlichkeit bei 8-Bit-Computern auf der Basis des U880-/Z80-/8080-Prozessors.

Neben anderen Möglichkeiten ist die Methode der „gepokten“ Assemblerprogramme ein Weg, der aus einer dBASE-Kommandodatei (CMD-Datei) heraus das Problem löst.

Die Umwandlung von (Assembler-)COM-Files in INLINE-POKEs ist aus Kompatibilitäts- und aus Einheitlichkeitsgründen für die dBASE Version 2.4x eine wirkungsvolle Methode (Alternative zum LOAD-Kommando). Vor jeder Abarbeitung eines „gepokten“ Unterprogramms sollte dasselbe immer wieder auf seine „Ladeadresse“ gebracht werden, da dBASE Version 2.4x z. B. den Speicher ab 0A400H gelegentlich selbst temporär auch neben dem SORT-Kommando nutzt. Erfahrene Programmierer können z. B. im Unterprogramm die BDOS-Beginnadresse entsprechend heruntersetzen, wodurch die Assemblerroutinen dann geschützt sind /2/. Das nachfolgende **Programm YPOKE.PAS** zeigt, wie man aus einem Assemblerprogramm, das als COM-File vorliegt, eine dBASE-Kommandodatei aus POKE-Befehlen generiert. Als Vorbereitung ist ein im 8080-/Z80-Code geschriebenes Assembler-Unterprogramm folgendermaßen in eine COM-Datei (eventuell Dateityp OVL oder OVX) zu überführen:

```
B>A:M80 <filename_REL>,<<^E>>
<filename_PRN>=<filename_MAC>/X/
M<CR>
B>A:L80 /P: <Adr_HEX>,<filename_
REL>/E<CR>
B>A:POWER SAVE <filename_COM>.
<type><<^E>>
  <Adr_HEX>  <anz><CR>
```

```
/~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\
A>WS<CR>
<<N>>                    <<B:POKEX.MAC<CR>>>
        .z80
        Title Testbeispiel fuer YPOKE
; POKEX.MAC
; Ladeadresse wird:   /P:A400  --->  41984
        .phase 0A400h
POKEX:: ld de,txt
        ld c,9
        call 5
        ret
txt:defb 12,13,10,10,134
    defb 'Testprogramm fuer YPOKE.PAS'
    defb 132,13,10,10
    defb 'Aufruf von dBASE II aus:'
    defb 13,10,9,9,134
    defb '. DO POKEX.CMD',132,13,10
    defb 9,9,134,'. CALL',132,24h
    .dephase
    end
<<^KD>>                                <<X>>
A>B:<CR>
B>A:M80 POKEX,POKEX=POKEX/Z/X/M/C<CR>
B>A:CREF80 LST:=POKEX.CRF<CR>
\_________________________________________/
```

Bild 1

```
/~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
A>L80 /P:A400,B:POKEX/E<CR>
A>DBASE<CR>
. SET CALL TO 41984<CR>
. CALL<CR>
* * * * * * * * * * * * * * * *


Testprogramm fuer YPOKE.PAS

Aufruf von dBASE II aus:
                  . DO POKEX.CMD
                  . CALL
. QUIT<CR>
A>_
\__________________________________
```

***Bild 2** Testlauf des Assemblerunterprogramms: POKEX.MAC*

Nach dem Übersetzen mit M80 muß die REL-Datei mit L80 auf die gewünschte Adresse gelinkt und geladen werden.

Die Abspeicherung erfolgt dann am besten mit POWER, das selbst nur die Adressen von 100H bis 3CFFH belegt. Der Wert für <anz> kann aus der Bildschirmanzeige des L80 entnommen und in die entsprechende dezimale Anzahl von Blöcken zu je 128 Byte für <anz> umgerechnet werden.

Die so gewonnene Maschinencodedatei <filename_COM>.<type> ist die Ausgangsbasis für das Programm YPOKE.COM, das aus dem Inhalt dieser Datei dBASE-II-POKE-Befehle für die explizit einzugebende Adresse <Adr_HEX> fortlaufend generiert.

Zur Demonstration das Programm in Bild 1 zum Aufruf mit CALL von dBASE II aus.

Vor der Generierung der POKE-Befehle sollte das Assemblerunterprogramm ausgetestet werden, indem es mit dem L80 aus der REL-Datei auf die geplante Ladeadresse gelinkt und danach mit dem dBASE-Aufruf und dem CALL-Kommando ausgeführt wird (siehe Bild 2).

Nach erfolgreichem Test wird dann mit:

```
A>L80 /P:A400,B:POKEX/E<CR>
A>POWER SAVE B:POKEX.OVX A400
1<CR>
```

das Unterprogramm ab der Adresse 0A400H bis 0A47FH als COM-File zur Weiterverarbeitung durch das Programm YPOKE abgespeichert (siehe Bild 3, Seite 308) (Link-80 Data A400 A466 < 102>).

In Bild 4 (Seite 308) ist der unmittelbar so nutzbare PASCAL-Quelltext des Programms YPOKE.PAS dargestellt.

Literatur

/1/ Meinhardt, I.: Arithmetik-Modul für REDABAS. Neue Technik im Büro, Berlin 31 (1987) 5, S. 143

/2/ Geiler, J.: Nachladbare Gerätetreiber für Personal- und Bürocomputer. Mikroprozessortechnik, Berlin 1 (1987) 6, S. 179

⊠ KONTAKT ☏

TU Dresden, Sektion Wasserwesen, Bereich WE; Tel. 2326118

TERMINE

Nationales Kolloquium „Computer und Gesellschaft“

WER? Bezirksverband Suhl der KDT, Wissenschaftssektion Computer- und Mikroprozessortechnik im Fachverband Elektrotechnik und Gesellschaft für Informatik, Fachsektion Informatik und Gesellschaft

WANN? 1. und 2. Dezember 1988

WO? Suhl

WIE? Teilnahmewünsche an BV der KDT Suhl, Straße der Opfer des Faschismus 29, Suhl, 6000; Tel. 22494

```
B>A:YPOKE B:POKEX.OVX   B:POKEX<CR>
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 [Assembler-]COM-File ---> DBASE-INLINE-POKEs
 ==== 8ØØØ ======== V.m=3.Ø =================

 B>A:M8Ø <filename_REL>,<filename_PRN>=<filename_MAC>/M<CR>
 B>A:L8Ø /P:<Adr_HEX>,<filename_REL>/E<CR>
 B>A:POWER SAVE <filename_COM>.<type>  <Adr_HEX>  <anz><CR>

        B>A:YPOKE B:POKEX.OVX   B:POKEX.CMD

 aktuelles Datum (tt.mm.jj) :Ø2.Ø3.88

 4-stellige hexadezimale Ladeadresse (<Adr_HEX>): A4ØØ
* B:POKEX.CMD vom: Ø2.Ø3.88
POKE 41984,17,9,164,14,9,2Ø5,5,Ø,2Ø1,12,13,1Ø,1Ø;
,134,84,1Ø1,115,116,112,114,111,1Ø3,114,97,1Ø9,1Ø9,32,1Ø2;
,117,1Ø1,114,32,89,8Ø,79,75,69,46,8Ø,65,83,132,13;
,1Ø,1Ø,65,117,1Ø2,114,117,1Ø2,32,118,111,11Ø,32,1ØØ,66
POKE 42Ø42,65,83,69,32,73,73,32,97,117,115,58,13,1Ø;
,9,9,134,46,32,68,79,32,8Ø,79,75,69,88,46,67;
,77,68,132,13,1Ø,9,9,134,46,32,67,65,76,76,132;
,36,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
POKE 421ØØ,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
SET CALL TO 41984
RETURN

Das POKEd Unterprogramm steht ab:   41984 (A4ØØh)
                             bis:   42111 (A47Fh)
```

Bild 3 Protokoll eines YPOKE-Laufes

Bild 4 Generierungsprogramm: YPOKE.PAS (erzeugt dBASE-II-POKEs aus einem COM-File)

```
  1:  Ø PROGRAM YPOKE;
  2:  Ø (*********************************************************)
  3:  Ø (* Fuer eine [Assembler-]COM-Datei wird eine POKE-Be-    *)
  4:  Ø (* fehlsfolge fuer dBASE II erzeugt, die das Programm    *)
  5:  Ø (* als INLINE-Code auf die LINK-Adresse, z.B. /P:A4ØØ    *)
  6:  Ø (* (41984), uebertraegt.                                  *)
  7:  Ø (* Das Programm erfordert die TURBO-Version 3.ØØx.        *)
  8:  Ø (*********************************************************)
  9:  Ø (*$V- *)
 1Ø:  Ø CONST K1='POKE '; K2='SET CALL TO '; K3='RETURN';
 11:  Ø TYPE setofchar = SET OF CHAR;
 12:  Ø      hilftext  = STRING[8Ø];
 13:  Ø VAR  f: FILE;  (* INPUT : COM-Datei als Untyped File *)
 14:  Ø      p: TEXT;  (* OUTPUT: dBASE-CMD-Datei *)
 15:  Ø      i,j,k,m,nrblocks : INTEGER;
 16:  Ø      Adr1,Adr     : REAL;          X          : CHAR;
 17:  Ø      Adress       : STRING[5];     Adresse    : STRING[6];
 18:  Ø      Name1,Name2: STRING[14];     Befehl     : STRING[3];
 19:  Ø      Datum        : STRING[8];
 2Ø:  Ø      Puffer       : ARRAY[Ø..127] OF BYTE;
 21:  Ø      Zeile,phase: hilftext;
 22:  Ø (*-------------------------------------------------------*)
 23:  Ø (* Quelle von Function lieszchn, Procedure lieswort:     *)
 24:  Ø (* Albuschat, F.: Eingabe unter Turbo-Pascal             *)
 25:  Ø (* Muenchen: Francis-Verlag, mc (1987)2, S. 62           *)
 26:  Ø FUNCTION lieszchn(zchnmenge:setofchar):CHAR;
 27:  Ø VAR ch: CHAR;
 28:  Ø     ok: BOOLEAN;
 29:  Ø BEGIN
 3Ø:  1   REPEAT
 31:  2     read(kbd,ch);
 32:  2     ok:=ch IN zchnmenge;
 33:  2     IF NOT ok THEN Write(chr(7))
 34:  2   UNTIL OK;
 35:  1   Write(ch);
 36:  1   lieszchn:=ch
 37:  1 END;
 38:  Ø
 39:  Ø PROCEDURE lieswort(zchnmenge:setofchar; maxlaenge:INTEGER;
 4Ø:  Ø                       VAR str:hilftext);
 41:  Ø VAR str1          : STRING[1];
 42:  Ø     strtemp       : STRING[8Ø];
 43:  Ø BEGIN
 44:  1   zchnmenge:=zchnmenge;
 45:  1   str1:=' ';
 46:  1   strtemp:='';
 47:  1   REPEAT
 48:  2   IF length(strtemp)=Ø
 49:  2        THEN str1[1]:=lieszchn(zchnmenge+[chr(13)])
 5Ø:  2        ELSE  IF length(strtemp)=maxlaenge
 51:  2          THEN str1[1]:=lieszchn([chr(13),chr(8)])
 52:  2          ELSE str1[1]:=lieszchn(zchnmenge+[chr(13),chr(8)]);
 53:  2   IF str1[1] IN zchnmenge THEN strtemp:=concat(strtemp,str1)
 54:  2        ELSE IF str1[1]=chr(8) THEN BEGIN
 55:  3                              Write(' ');
 56:  3                              Write(chr(8));
 57:  3                              delete(strtemp,length(strtemp),1) END
 58:  2   UNTIL str1[1]=chr(13);
 59:  1   IF length(strtemp)<>Ø THEN str:=strtemp
 6Ø:  1 END;
 61:  Ø (* ----------------------------------------------------- *)
 62:  Ø FUNCTION HEXIN: REAL;
 63:  Ø CONST hexset: ARRAY[Ø..15] OF CHAR='Ø123456789ABCDEF';
 64:  Ø VAR (* phase: hilftext;    <--- im Hauptprogramm          *)
 65:  Ø     XP   : ARRAY[1..4] OF BYTE;  i,j : INTEGER;
 66:  Ø BEGIN
 67:  1   Writeln;
 68:  1   REPEAT
 69:  2     Write(' 4-stellige hexadezimale Ladeadresse'+
 7Ø:  2       ' (<Adr_HEX>): ',phase:4,#8#8#8#8);
 71:  2     lieswort(['Ø'..'9','A'..'F','a'..'f'],4,phase)
 72:  2   UNTIL length(phase)=4;
 73:  1   FOR i:=1 TO 4 DO BEGIN
 74:  2     phase[i]:=upcase(phase[i]);j:=Ø;
 75:  2     WHILE hexset[j]<>phase[i] DO j:=succ(j);
 76:  2     XP[i]:=j;
 77:  2   END;
 78:  1   HEXIN:=4Ø96.Ø*XP[1]+256.Ø*XP[2]+16.Ø*XP[3]+XP[4]
 79:  1 END;
 8Ø:  Ø (*--------------------  Hauptprogramm  ------------------*)
 81:  Ø BEGIN
 82:  1   CBREAK:=TRUE;phase:='A4ØØ';
 83:  1 (* CTRL-Steuerzeichen werden akzeptiert  --> <<^C>> BREAK *)
 84:  1   clrscr;
 85:  1    Writeln(#1Ø#1Ø#134+
 86:  1    '[Assembler-]COM-File ---> DBASE-INLINE-POKEs'#132);
 87:  1    Writeln(
 88:  1    ' ==== 8ØØØ ======== V.m=3.Ø =================');Writeln;
 89:  1    Writeln(' B>A:M8Ø <filename_REL>,<filename_PRN>='+
 9Ø:  1    '<filename_MAC>/M<CR>');
 91:  1    Writeln(' B>A:L8Ø /P:<Adr_HEX>,<filename_REL>/E<CR>');
 92:  1    Writeln(' B>A:POWER SAVE <filename_COM>.<type>'+
 93:  1    '  <Adr_HEX>  <anz><CR>');                  GotoXY(1,1Ø);
 94:  1    Writeln(#134'          B>A:YPOKE <filename_COM> '+
 95:  1    ' <filename_CMD><CR>'#132#13#1Ø);
 96:  1    IF ParamCount<2 THEN BEGIN
 97:  2      Writeln('*** Ein- und Ausgabedatei in der Aufruf'+
 98:  2      'zeile angeben'#13#1Ø'===> Noch einmal !'); HALT END
 99:  1                    ELSE BEGIN
1ØØ:  2      Name1:=ParamStr(1); Name2:=ParamStr(2);
1Ø1:  2      IF POS('.',Name1)=Ø THEN Name1:= Name1+'.OVX';
1Ø2:  2      IF POS('.',Name2)=Ø THEN Name2:= Name2+'.CMD';
1Ø3:  2      (*$I-*) ASSIGN(f,Name1); RESET(f); (*$I+*)
1Ø4:  2      IF IORESULT<>Ø THEN BEGIN
1Ø5:  3        Writeln ('*** Eingabedatei ',Name1,
1Ø6:  3         ' nicht gefunden --> CANCEL !');HALT
1Ø7:  3      END;
1Ø8:  2    END;
1Ø9:  1    nrblocks:=filesize(f);
11Ø:  1    IF nrblocks=Ø THEN BEGIN
111:  2    Writeln('Eingabedatei ist leer ---> CANCEL'); HALT END;
112:  1    ASSIGN(p,Name2); REWRITE(p);
113:  1    GotoXY(2Ø,1Ø);CLREOL;(* Zeile 1Ø loeschen ab Pos. 2Ø *)
114:  1    GotoXY(2Ø,1Ø);  Writeln(Name1,'    ',Name2,#132);
115:  1    Write(#13#1Ø' aktuelles Datum (tt.mm.jj) :');
116:  1    Readln(Datum);
117:  1    Adr:=HEXIN;          Adr1:=Adr;         Writeln;
118:  1    J:=Ø;
119:  1    zeile:='* '+Name2+' vom: '+Datum;
12Ø:  1    FOR i:=1 TO nrblocks DO BEGIN
121:  2      blockread(f,Puffer,1);
122:  2      m:=Ø;
123:  2      REPEAT
124:  3        k:=Puffer[m];
125:  3        str(k,Befehl);
126:  3        IF (J MOD 15 =Ø) THEN BEGIN
127:  4          IF (J MOD 6Ø =Ø) THEN BEGIN
128:  5                               Writeln(p,zeile); Writeln(zeile);
129:  5                               str(adr:5:Ø,adress);
13Ø:  5                               Zeile:=k1+Adress+','+Befehl;
131:  5                               J:=J+2 END
132:  4                            ELSE BEGIN
133:  5                               Zeile:=Zeile+';';
134:  5                               Writeln(p,Zeile); Writeln(Zeile);
135:  5                               Zeile:=','+Befehl END
136:  4          END ELSE Zeile:=Zeile+','+Befehl;
137:  3        Adr:=Adr+1;
138:  3        m:=succ(m);J:=succ(J)
139:  3      UNTIL m=128;
14Ø:  2    END;
141:  1    Writeln(p,Zeile); Writeln(Zeile);
142:  1    STR(Adr1:5:Ø,Adress); Zeile:=K2 + Adress;
143:  1    Writeln(p,Zeile); Writeln(Zeile);
144:  1    Writeln(p,K3); Writeln(K3);
145:  1    CLOSE(p);
146:  1    Adr:=Adr-1;
147:  1    Write(#13#1Ø'Das POKEd Unterprogramm steht ab: ',Adr1:7:Ø,
148:  1      ' (',phase,'h)'+
149:  1        #13#1Ø'                              bis: ', Adr:7:Ø,
15Ø:  1      ' (');                         STR(Adr:5:Ø,Adress);
151:  1    FOR I:=1 TO 5 DO IF Adress[I]=' ' THEN Adress[I]:='Ø';
152:  1    VAL(Adress,Adr1,I);
153:  1    IF Adr1>Maxint THEN BEGIN
154:  2                           Adr1:=-65536.Ø+Adr1;
155:  2                           STR(Adr1:6:Ø,Adresse);
156:  2                           VAL(Adresse,J,I) END
157:  1                    ELSE BEGIN
158:  2                           STR(Adr1:5:Ø,Adress);
159:  2                           FOR I:=1 TO 5 DO
16Ø:  2                           IF Adress[I]=' ' THEN Adress[I]:='Ø';
161:  2                           VAL(Adress,J,I) END;
162:  1    IF I=Ø THEN  INLINE($2A/J/$CD/$4AF); Writeln('h)');
163:  1    REPEAT UNTIL keypressed
164:  1 END.
```

TERMINE

17. Jahrestagung
„Grundlagen der Modellierung und Simulationstechnik"

WER? Wissenschaftlich-Technische Gesellschaft für Meß- und Automatisierungstechnik der KDT
WANN? 7. bis 9. Dezember 1988
WO? Rostock
WAS?

- Simulationsmethoden und -techniken
- Mathematische Modellierung von Umweltproblemen sowie beim Schaltkreisentwurf
- Simulation auf 16-Bit-Mikrorechnern

WIE? Teilnahmewünsche bitte an Kammer der Technik, Präsidium, WGMA, Clara-Zetkin-Straße 115–117, Berlin, 1086

Müller

Mikroprozessor-Nachnutzungsleiterplatten

Volkmar Tetzlaff
Ingenieurbetrieb für die Anwendung der Mikroelektronik Erfurt

Einführende Bemerkungen

Auf der Grundlage des Beschlusses des Politbüros des ZK der SED vom 26. 6. 1979 zur beschleunigten Entwicklung und Anwendung der Mikroelektronik in der Volkswirtschaft der DDR wurden im Jahre 1980 in allen Bezirken die *Beratungs- und Informationsstellen Mikroelektronik* (BIS) geschaffen. Mit der planmäßigen Weiterentwicklung dieser BIS verlagerte sich in den Jahren 1983–1985 der Schwerpunkt von der beratenden Funktion zu einer Einrichtung mit Ingenieurbürocharakter, so daß folgerichtig im Jahre 1986 die namentliche Umbenennung der BIS Erfurt in den Ingenieurbetrieb für die Anwendung der Mikroelektronik (IfAM) erfolgte.
Der IfAM Erfurt ist ein Partner der Klein- und Mittelbetriebe bei der territorialen Rationalisierung.

Leistungsangebote

- Kostenlose Beratung bei der Einsatzvorbereitung der Mikroelektronik
- Entwicklung und Anpassung von Software auf allen verfügbaren 8-Bit- und 16-Bit-Mikrocomputern, wie **BC 5120, BC 5130, PC 1715 (W), AC 7100, AC 7150, EC 1834**, Schneider-**PC (1512, 1640)**, **XT**- und **AT**-kompatiblen Computern, **P 8000**, für die Betriebssysteme **SCP, SCP 1700, UDOS, CP/M, CP/M-86, MS-DOS, DCP, UNIX**
- Realisierung von Mikrorechnern für die MSR- und Automatisierungstechnik (Hard- und Software)
- CAD-unterstützte Herstellung von Leiterplattenlayouts (Lp-Fertigungsdokumentation nach Kunden-2:1-Vorlagen); Fertigung von Musterleiterplatten (2-Ebenen-Leiterplatten in DKL-Qualität)
- Bereitstellung von Nachnutzungs-Leiterplatten (unbestückt mit dazugehöriger umfangreicher Dokumentation, Aufbau- und Inbetriebnahme-Anleitung und Softwarekomponenten)
- Qualifizierungslehrgänge zu Mikroprozessoren (U 880, EMR U 882...)

– Programmiersprachen (Assembler [U 880, EMR U 882], BASIC, PASCAL)
– Universalsoftware (Textprozessor, REDABAS, Tabcalc)
– Mikrorechnerbetriebssysteme aller Leistungsklassen (SCP, DCP, MS-DOS).

Nachnutzungsleiterplatten im Überblick

Mit dem Aufbau eines leistungsfähigen Leiterplatten-Entwurfszentrums im IfAM Erfurt, wo auch ständig Kundenaufträge bearbeitet werden können, wurden die Grundlagen für die Bereitstellung einer Vielzahl von Nachnutzungs-Leiterplatten (NN-Lp) geschaffen. Bei der Entwicklung und Bereitstellung dieser NN-Lp in ausreichender Zahl und Qualität wird auf deren universellen multivalent einsetzbaren Charakter Wert gelegt. Dem technischen Trend folgend wird sich dabei der Schwerpunkt in den nächsten Jahren deutlich von den K-1520-OEM-Ergänzungsbaugruppen zu Erweiterungskarten moderner 16-Bit-PCs, wie EC 1834/35, Schneider-PC, XT- und AT-kompatiblen Computern sowie zu universellen Prozeßkoppelmodulen, die sich an diese Rechner anschließen lassen, verlagern.
Der IfAM Erfurt ist dabei ständig an der Zusammenarbeit mit Partnern aus der Industrie und wissenschaftlichen Einrichtungen bei der Entwicklung von nachnutzbaren Lösungen interessiert. Damit universelle Ergänzungsbaugruppen schnell einem breiten Interessentenkreis zugänglich gemacht werden können, übernimmt der IfAM Erfurt auch auf vertraglicher Basis komplette Lösungen, die technologisch überarbeitet als NN-Lp angeboten werden können.
Das Nachnutzungssortiment des IfAM Erfurt umfaßt z. Z. folgende Komplexe:

- PC-Ergänzungsbaugruppen für moderne 16-Bit-Computer
- K-1520-kompatible OEM-Ergänzungsbaugruppen
- Universelle Prozeßkoppelmodule
- sonstige Nachnutzungsbaugruppen für Emulation, Entwicklungsmodule, Einplatinencomputer, EPROM-Programmierung, Experimentierleiterplatten u. a.

Vorstellung der Nachnutzungsleiterplatten

PC-Ergänzungsbaugruppen
Rasterleiterplatten

PC-RK Raster-Lp zur universellen Verdrahtung
- vorgegebene Leiterzüge für 5P und GND
- freies Rasterfeld (2,54 mm)
- *lieferbar: ab 10/88*

nach Konsultation mit dem IfAM über Einsatzzweck und mögliche kooperative Entwicklung
PC-RK1 Raster-Lp zur universellen Verdrahtung
- 5P und GND sind als Gitterstruktur zwischen den Lötaugen auf L- und B-Seite der Lp ausgeführt
- freies Rasterfeld (2,54 mm)
- Lötstopmaske
- *lieferbar: ab 10/88*

Prozeß-Ein-Ausgabe

PC-BA Busanpassung
- 8 I/O-select-Signale
- 80% der Lp ist freies Rasterfeld (2,54 mm)
- *lieferbar: ab IV/88*

PC-UP Universelle Peripherie
- 16 AD-Kanäle (10 Bit)
- 2 DA-Kanäle (8 Bit)
- je 8 digitale I/O-Kanäle
- *lieferbar: ab IV/88*

PC-AIP 2 × Parallelinterface IFSP (Ein-/Ausgabe)
- z. B. Anschluß GST K 7067.12(15)
- 1 × 8-Bit-Ausgabeschnittstelle (open collector)
- *lieferbar: ab I/89*

PC-EE Erweiterungsbus Empfänger
- *lieferbar: ab IV/88*

PC-ES Erweiterungsbus Sender
- *lieferbar: ab IV/88*

PC-EB Erweiterungsbus Rückverdrahtung
- *lieferbar: ab IV/88*

Allgemeine Angaben
Abmessungen: alle Lp 360 mm × 107,62 mm (L × B)
Nennmaß ohne vergoldeten Kontaktkamm: 360 mm × 100 mm
Die benötigten Lp-Formate können vom Anwender selbst hergestellt werden. Auf jeder Lp sind die dazu notwendigen Markierungen vorhanden.
Empfohlene Lp-Formate:

EC 1834:	100 mm × 360 mm
	100 mm × 300 mm
	100 mm × 240 mm
XT:	100 mm × 338 mm
(Original:	106,7 mm × 333,5 mm)
AT:	100 mm × 338 mm
(Original:	121,9 mm × 333,5 mm
Schneider-PC:	100 mm × 338 mm

Bussteckverbinder:
– wahlweise direkt (**XT, AT, Schneider**)
– 2 × 31polig und 2 × 18polig
– Raster 2,54 mm, vergoldet 1,5 μm oder indirekt (**EC 1834**)
– 1 × 96polige Buchsenleiste **R96 F01**
Interfacesteckverbinder:
– Anschlußmöglichkeit für einen 9-, 15-, 25-, 37-, 50poligen D-Subminiatur-Gerätestecker für die zusätzliche **I/O**-Übergabe
Basismaterial:
– Epoxidglashartgewebe nach TGL 11651 **S2C**, 1,5 mm stark, zweiseitig, 35 μm Kupfer, durchk.

K-1520-kompatible OEM-Ergänzungsbaugruppen
Prozeß-Ein-/Ausgabe

AD-1520 Analog-Digital-Wandler /1, 2, 13/
- 16-Kanal-Analogeingabe für **K-1520-MR**
- Auflösung: 10 Bit/3 Digit
- Umsetzer: **C 571/C 520**
- Umsetzzeit: 25 μs (typisch)/50–160 pro s
- Eingangsspannungsbereich:

0 ... 5 V/–99 mV ... +999 mV
- Eingangsmultiplexer: U 105 D
- *lieferbar: sofort*

AB-1520 Prozeßausgabe /1, 2/
- Ansteuerung von Stellgliedern bzw. Verbrauchern mittels potentialfreier Relaiskontakte und/oder potentialbehafteter Transistorschaltstufen
- 8 Leistungsausgänge mit 2 Bestückungsvar.

a) 4 × 220 V, 1 A A; × 150 V, 1 A; Relais
b) 8 × Transistorschaltstufen
- 8 Leitungen als TTL-I/O-Portleitungen
- *lieferbar: sofort*

AZK-1520 Anzeigebaugruppe
- 8-Digit-7-Segment-Anzeigen (**VQE**)

Lp-Format: 140 mm × 60 mm
- *lieferbar: sofort (ohne Dokumentation)*

ZVE

ZVE-1520 (K 2521) Zentrale Verarbeitungseinheit
- **U 880, PIO, CTC**
- **ROM: 3 KByte (3 × U 555)**
- **RAM: 1 KByte (8 × U 202)**
- *lieferbar: sofort*

ZVE-64 Zentrale Verarbeitungseinheit
- vollständig gepuffertes Rechnersystem
- analog K 2521
- ohne Mehrrechnerkopplung
- Einsatz als 64K-RAM-Rechner ist möglich
- ROM: maximal 16 KByte (**2 × U 2716 ... U 2764**), ein-/ausschaltbar
- RAM: 64 KByte (**8 × U 2164**)
- **1 × SIO, 1 × CTC, 1 × PIO**
- **V.24**-Treiber für **r×d, t×d** asynchron
- Anschluß über Anwenderbus-Steckverbinder
- Betriebsspannungen: **5P(1A), 12P/12N** (je 50 mA)
- Standard-Systembus
- *lieferbar: ab 9/88*

Speicher

OFS (K 3620) Operativ-Festwertspeicher
- **ROM:** 6 KByte (**6 × U 555**)
- **RAM:** 2 KByte (**16 × U202**)
- *lieferbar: sofort*

OPS 64K Operativ-Speicher /1, 2/
- RAM: 64 KByte-**dRAM (32 × U216)**
- Organisation: 64 KByte (8 Bit) oder 32 **KWord**
- Refresh: extern
- *lieferbar: sofort*

16K-CMOS-RAM Operativ-Speicher /1, 3/
- RAM: 16-KByte-**CMOS-RAM (32 × U 224)**
- Pufferung: **3 × NC**-Knopfzellen
- *lieferbar: sofort*

1M-RAM RAM-Diskette
- RAM: 256 KByte (**32 × U 2164**)/1 MByte (**32 × U 21256**)
- Bankbetrieb möglich
- *lieferbar: ab 9/88*

PFS-I (K 3820) Programmierbarer Festwertspeicher
- **ROM:** 16 KByte **(16 × U 555)**
- *lieferbar: sofort*

PFS-II Programmierbarer Festwertspeicher
- wahlweise ROM/RAM bestückbar
- **ROM:** 32 KByte **(16 × U 2716)** oder 64 KByte **(16 × U 2732)** oder
- **RAM:** 32 KByte **(16 × UL 6516)**
- **U 2716** auf Leiterplatte programmierbar!
- Standard-Systembus
- *lieferbar: ab IV/88*

Peripherie

KK Koppelkarte /1, 3, 4/
- Kopplung zweier **K-1520**-Mikrorechner
- gemeinsamer Koppel-RAM: 8 KByte **(16 × U 214)**
- Zugriff auf Koppel-RAM von beiden Rechnern ohne Priorität möglich
- Steuerung erfolgt über **WAIT**-Leitungen
- Asynchrone Arbeit beider Rechner gegeben
- *lieferbar: sofort*

SIO 1/2 Serielles Interface /3/
- Schnelles synchrones serielles Interface
- maximale Übertragungsrate: 500 KBaud
- maximale Leitungslänge: 100 m (Koax, 75 Ohm)
- freie Adressierbarkeit: **AB2 – AB7**
- *lieferbar: sofort*

SIO 3 Synchrones/asynchrones serielles Interface
- 6 Eingangs-/Ausgangskanäle **(3 × SIO)**
- 6 Zeitgeberausgänge
- 8 Triggereingänge **(2 × CTC)**
- *lieferbar: sofort*

V 24 K 8021 V.24-Interface
- zweifach serielles Interface
- Übertragungsrate: 110 Baud–9,6 KBaud
- *lieferbar: sofort*

VIS 3 Videoansteuerung mit **GDC U 82720** /5/
- 16 Farben, Auflösung: 384 × 288 Pixel
- Zooming: 1 ... 16, Bildfenster: 2
- Panning: vertikal und horizontal
- komplette Ansteuerung für Farbmonitor **MON K 7226.X0** mit
  - Bildspeicherung
  - Erzeugung und Aufbereitung der Videosignale und des Synchronsignales
  - Kaskadierung mehrerer **VIS 3**
  - Betrieb eines Lichtstiftes
  - Bus-Interface einschließlich DMA-Zugriff für **K-1520**-Systeme
- *lieferbar: sofort (Software über AdW/ZWG Berlin)*

Floppy-C Floppy-Disk-Controller-Leiterplatte
- äquivalent **K 5126**
- **FDC: U 8272, DMA: I 8257, PIO: U 855**
- Anschluß: 4 FD-Laufwerke, 5.25 oder 8 Zoll
- *lieferbar: ab 9/88*

Ergänzungsbaugruppen

NT-1520 Stabilisierte Stromversorgung /1, 2/
- 5P/2,5 A (MAA 723)
- 5N/0,5 A (KU 607) aus 12N abgeleitet
- 12P/1 A (MA 7812)
- 12N/0,5 A (MA 7812)
- Lp: S1C, NDKL
- *lieferbar: sofort*

RV 5 Rückverdrahtungs-Leiterplatte
- 5 Steckplätze für Bussteckverbinder
- Lp-Format: 110 mm × 80 mm
- *lieferbar: sofort*

RV 1 Rückverdrahtungs-Leiterplatte
- 11 Steckplätze für Bussteckverbinder
- Lp-Format: 230 mm × 80 mm
- *lieferbar: sofort*

Allgemeine Angaben

Abmessungen:
- alle Lp 215 mm × 170 mm (L × B)

Busanschluß:
- über 2 × 58polige Steckverbinder
- indirekt TGL 29331/03
- **K-1520**-kompatibel

Basismaterial:
- Epoxidglashartgewebe nach TGL 11651, **S2C**, 1,5 mm stark, zweiseitig

Universelle Prozeßkoppelmodule

UPK1 ZVE
- Kalibrierung, Überwachung und Glättung der Meßwerte
- Kanalfolge, Grenzwerte, Masken und Meßwerte können über das serielle Interface vom Steuerrechner definiert bzw. abgerufen werden
- **EMR: U 8840**
- **ROM: U 2732**
- **RAM: U 6516**
- **serielle Schnittstelle (V.24 bzw. LWL)**
- *lieferbar: ab 1989*

UPK2 Analog-Digital-Wandler
- 32 Kanäle (−5 V ... +5 V), **C 571** oder **C 574**
- Sample & Hold möglich
- *lieferbar: ab 1989*

UPK3 Digital-Analog-Wandler
- 4 Kanäle mit je 10 Bit **(C 565)** oder 8 Bit **(C 560)**
- UXI-Wandler möglich (0 ... 20 mA)
- *lieferbar: ab 1989*

UPK4 Digital-Ein-/Ausgabe
- 8 Eingänge mit Überlastschutz (12 ... 30 V)
- 8 Ausgänge (Transistoren 12 ... 30 V/0,5 A)
- Potentialtrennung durch Optokoppler
- *lieferbar: ab 1989*

UPK0 Systembus
- *lieferbar: ab 1989*

Allgemeine Angaben

Haupteinsatzgebiet:
- Meß-, Steuer- und Regelungstechnik

Verwendung:
- Schnittstelle zwischen technologischem Prozeß und beliebigem Steuerrechner **(PC 1715, EC 1834)**
  - Betriebsspannungen: 5P, 15P, 15N
  - Aufbauvarianten:
  - maximal je 2 E/A-Module **(UPK2 ... 4)**
  - interne softwaremäßige Anpassung an die Hardwarekonfiguration erfolgt automatisch

Abmessungen:
- alle Lp 170 mm × 95 mm

Basismaterial:
- Epoxidglashartgewebe nach TGL 11651, S2C, 1,5 mm stark, zweiseitig, 35 μm Kupfer, durchkontaktiert

Sonstige Nachnutzungsbaugruppen

Emulation

EMU-I U-880-Mikrorechner-Inbetriebnahmemodul /6/
- besteht aus 4 Lp:
  ZVE-1520, OPS 64K, V.24 und **EMU-I**
- Steuerung: über V.24 vom PC
- Umfangreiche Software
- Adreßraum des Anwendersystems: 0 ... DFFFH
- *lieferbar: sofort*

E-Modul Emulator-Lp für **EMR U 882/884**
- Software und Dokumentation über TU Karl-Marx-Stadt/IT, Dr. Franke
- *lieferbar: sofort*

Entwicklungsmodule

16-Bit-Entwicklungsmodul
- Einarbeitung in die Hard- und Softwareproblematik von 16-Bit-MP (U 8000)
- Charakteristik:
  - 16-Bit-Mikroprozessor **(U 8002)**
  - **2-KWord-EPROM** (erweiterbar auf 8 KWord)
  - **32-KWord-RAM**
  - zweifach serielles Interface (V.24)
  - 48 programmierbare I/O-Leitungen mit Handshakesteuerung **(PIO)**
  - 4 programmierbare 8-Bit-Zähler/Zeitgeber **(CTC)**
- Lieferumfang: 3 Leiterplatten und **1 × IC (U 8002)**
  - **CPU, I/O, OPS 64K**
- weitere benötigte Leiterplatten:
  - **2 × PFS-I(K 3820)**
- *lieferbar: sofort*

Einplatinen-Computer

EKR 1 Einkartenrechner /1, 7/
- für Steuerung und Regelung
- **EMR: U 884 X**
- **ROM:** 4 KByte **(2 × U 2716)**
- **AD**-Umsetzer: **C 571/C 570**, 10 Bit/8 Bit
- **DA**-Wandler: **C 5650/C 5658**
- digitale I/O-Leitungen
- Inbetriebnahme und Test über 39poligen Steckverbinder möglich
- *lieferbar: sofort*

EPR 1 Einplatinenrechner /1, 3/
- **MP: U 880**
- **RAM:** 4 KByte **(8 × U 224** oder **U 214)**
- **ROM:** 8 KByte **(4 × U 2716)**
- **2 × PIO:** 4 PIO-Ports **(32 I/O-Leitungen)**
- **2 × CTC:** 8 Triggereingänge, 6 Zeitgeberausgänge
- **2 × SIO:** 4 **SIO**-Ports mit Steuerleitungen
- Bussystem: nicht nach außen geführt
- Lp-Format: 230 mm × 125 mm
- *lieferbar: sofort*

EPR2 Einplatinenrechner
- Software: kompatibel zum **ZX-Spectrum/ZX-128**
- **MP: U 880**
- **RAM:** 64 KByte/128 KByte (8/16 × U 2164)
- **ROM:** 16 KByte (2 × U 2764)
- Monitoranschluß: RGB (Farbe)
- **TV**-Anschluß: HF (sw)
- Joystickanschluß
- Stromversorgung und **Tastatur** müssen vom Anwender realisiert werden
- Lp-Format: 300 mm × 200 mm
- *lieferbar: ab 10/88*

Sonstige Leiterplatten

EMR-I Experimentierleiterplatte /8/
- für Einchipmikrorechner **U 882X/U884X**
- vorgegebene Minimalkonfiguration:
  - **EMR**
  - **EPROM**
  - Taktgenerator
- Rasterfeld zur individuellen Verdrahtung
- EMR-Signale über Steckverbinder
- *lieferbar: sofort*

VQC 10 Alphanumerische Zeichendarstellung /9/
- 16 Stellen **(4 × VQC 10)**
- Ansteuerung: durch **EMR** (z. B. **EMR-I**) möglich
- **Lp**-Format: 160 mm × 60 mm
- *lieferbar: sofort (ohne Dokumentation)*

EPROM-Programmierung

EPROM 1 EPROM-Programmierzusatz /10, 11/
- für **PC 1715/K 1520**
- EPROM-Typen: **2708 ... 27256**
- Zusatz-Lp (EPROM1-Z gehört zum Lieferumfang) für 28polige (40-polige) Schwenkhebelfassung, extern anschließbar
- Lp-Formate: **K 1520** oder 215 mm × 150 mm **(PC 1715)**
- *lieferbar: sofort*

EPROM 1-W EPROM-Programmierzusatz /12/
- für **PC 1715/K 1520**
- **EPROM**-Typen: **2716 ... 27513** (16-Bit-EPROM in Vorbereitung)
- Programmierarten: Fast, Standard
- **Lp**-Format: Konfigurierung durch Anwender
- *lieferbar: ab IV/88*

EPROM 2 EPROM-Programmierzusatz /12/
- Centronics-Interface **(MS-DOS-PC)**
- EPROM-Typen: **2716 ... 27256** (weitere Typen in Vorbereitung)
- Programmierarten: Fast, Standard
- *lieferbar: ab IV/88*

Allgemeine Lieferbedingungen

Lieferumfang:
- Nachnutzungsdokumentation
- 1 Satz/1 Stück unbestückte Leiterplatte (Leiterplatten sind industriell gefertigt und durchkontaktiert)
- weitere Leiterplatten auf Anfrage lieferbar
- Software (auf Datenträger des Kunden)

Bestellung:
- Schriftlich (formlos) an Kontaktadresse unter Angabe von Leiterplattentypen, Stückzahl

Vertragsgestaltung:
- Nach Eingang der Bestellung erhält der Kunde ein schriftliches Vertragsangebot mit einer detaillierten Aufstellung des Leistungsumfanges, der Lieferzeit und der Preise.

Versand:
- Nach Eingang des unterschriebenen Vertrages im IfAM Erfurt erfolgt termingemäß der Versand der Ware per Post (Selbstabholung kann vereinbart werden).

Rechnungslegung:
- Erfolgt nach Versand der Ware

Persönliche Konsultationen sind nach **telefonischer** Voranmeldung möglich.

Fortsetzung auf Seite 312

RVS K 1840

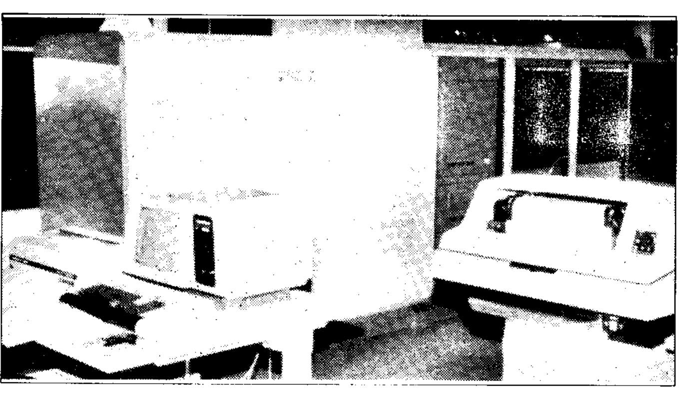


Foto: Weiß

Zur Leipziger Frühjahrsmesse offerierte das Kombinat Robotron das Rechnersystem mit virtuellem Speicher (RVS) K 1840. Es besitzt eine 32-Bit-Architektur, ein Prozeßkonzept, ein virtuelles Speicherkonzept mit Speicherverwaltungs- und Schutzmechanismen sowie einen Befehlssatz mit einer Vielzahl von Datentypen und Adressierungsarten. Der K 1840 verwendet das Cacheprinzip zum Bereitstellen von Befehlen, Adressen und Daten, besitzt Datensicherheitsmechanismen und Diagnosefähigkeit. Er ist kompatibel zur 16-Bit-SKR-Reihe K 1600, CM 3 und CM 4. Hauptanwendungsgebiete sind der Entwurf und die Simulation höchstintegrierter Schaltkreise und hochkomplexer Leiterplatten sowie komplexe Konstruktions-, Entwurfs- und Projektierungsarbeiten mit 2- und 3dimensionalen Objekten. Der K 1840 ist als Steuerrechner in durchgängigen CAD/CAM-Anwendungen und als Mehrnutzersystem einsetzbar. Er besitzt einen virtuellen Adreßraum von 4 GByte, der in den Systemraum für das Betriebssystem und den Prozeßraum für die Nutzer unterteilt ist. (Der Adreßraum von 4 GByte heißt virtuell, weil physisch nur ein RAM von einigen MByte vorhanden ist; dieser für den Nutzer scheinbar existierende RAM wird vom Rechner durch ständige Datenumlagerungen zwischen RAM und Massenspeicher erreicht.) Im Prozeßraum können durch jeden der maximal 16 Nutzer Datenstrukturen bis zur Größe von einem GByte erzeugt werden. Im Grundmodus besitzt der K 1840 einen Befehlssatz von 304 Befehlen. Die Datentypen sind im Integer-Format Byte (8 Bit; Bereich $-128 \ldots +127$ bzw. $0 \ldots 255$), Wort, Langwort, Quadwort und Oktavwort (128 Bit; Bereich $-2^{127} \ldots +2^{127}-1$ bzw. $0 \ldots 2^{128}-1$) sowie im Gleitkommaformat F-Format (32 Bit; Bereich $0{,}29 \cdot 10^{-38} \ldots 1{,}7 \cdot 10^{38}$), D-Format, G-Format und H-Format (128 Bit, Bereich $0{,}84 \cdot 10^{-4932} \ldots 0{,}59 \cdot 10^{4932}$). Maßnahmen zur Gewährleistung der Datensicherheit sind eine Fehlererkennungs- und Korrekturschaltung zur Korrektur von Ein- und Zweibitfehlern sowie einiger Mehrbitfehler, die Verwendung von Paritätsbits im Mikroprogrammspeicher sowie eine Speicherstützung zur Überbrückung von Netzausfällen (10 Minuten). Das Rechnersystem mit virtuellem Speicher besteht aus folgenden Baugruppen:

- Zentrale Verarbeitungseinheit (CPU)
- Gleitkommabeschleuniger
- Konsolsubsystem
- Interner Synchronbus
- SKRBUS-Adapter
- MSBUS-Adapter

Die diskrete **CPU** beinhaltet folgende Funktionsgruppen:

- Mikroprogrammsteuerung
- Rechenwerk
- Befehlspuffer
- Datencache
- Adreßübersetzungspuffer
- ISB-Steuerung
- Unterbrechungsbehandlung
- Taktversorgung.

Weiterhin enthält die CPU eine Echtzeituhr sowie eine gepufferte Uhr für Tageszeit und Datum.

Der **Hauptspeicher,** der mindestens 2 MByte betragen muß, wird in der Standardausführung mit 8 MByte ausgerüstet und ist bei Verwendung der 64-KBit-dRAMs in 2-MByte-Schritten auf 16 MByte aufrüstbar. Mit dem Einsatz des 256-KBit-dRAMs, dessen Massenproduktion im Kombinat Mikroelektronik vorbereitet wird, dürfte eine Vergrößerung des Hauptspeichers auf 64 MByte möglich werden. Im Speicher können Schreib- und Leseoperationen mit 32-Bit-Langworten und 64-Bit-Quadworten ausgeführt werden.

Der **SKRBUS-Adapter** verbindet den Internen Synchronbus mit dem asynchronen SKRBUS. Damit können alle peripheren Geräte für die Kleinrechner (SKR) auch an den K 1840 angeschlossen werden.

Der **MSBUS-Adapter** (MBA) verbindet den Internen Synchronbus mit dem MSBUS, der wiederum für den Anschluß der Plattenspeicher dient. An den Internen Synchronbus können zwei MSBUS-Adapter und an einen MSBUS-Adapter maximal 8 Plattenspeicher angeschlossen werden (der zweite MBA ist optional).

Als Massenspeicher kann erstens der **Winchester-Festplattenspeicher K 5502** von Robotron mit einer Speicherkapazität (formatiert) von 124 MByte und einer mittleren Zugriffszeit von 52,5 ms angeschlossen werden. Der Festplattenspeicher mit den Abmessungen 482 × 770 × 265 mm^3, einer Masse von 60 kg und einer Leistungsaufnahme von 550 VA wird über einen PM-Adapter (Masse: 45 kg; Leistungsaufnahme: 350 VA) an den MBA angeschlossen.

Zweitens können die **Wechselplattenspeicher CM 5404** oder **CM 5416** von ISOT (Bulgarien) mit 88 bzw. 174 MByte Speicherkapazität und einer mittleren Zugriffszeit von 30 ms Verwendung finden. Bei einem Volumen (einschließlich integriertem Adapter) von 990 × 820 × 1150 mm^3 und einer Masse von 351 kg nehmen sie 2 KVA Leistung auf. Als drittes Massenspeichermedium können **Magnetbandgeräte** vom Typ **CM 5306, CM 5308** und **CM 5309** von ISOT dienen. Die Speicherkapazität pro 0,5-Zoll-Magnetband beträgt bis zu 40 MByte. Die Magnetbandcontroller, die je vier Magnetbandgeräte ansteuern können, werden an den SKRBUS angeschlossen.

Als alphanumerisches Terminal dient standardmäßig das **Terminal K 8911.80.** Für Farbgrafikarbeit wird das **Interaktive Grafische Terminal K 8918** eingesetzt. Es kann über einen s/w- oder Farbmonitor sowie über ein Grafiktablett K 6405 verfügen. Auf dem Monitor können 640 × 480 Pixel in 16 Farben bzw. Graustufen angezeigt werden. Die Darstellung von alphanumerischen und grafischen Daten ist kombiniert möglich. An beide Terminals kann für Hardcopy ein Drucker K 6313/14 angeschlossen werden.

Die Terminals werden über den **Multifunktions-Kommunikationscontroller AMF18** mit dem SKRBUS gekoppelt. Der AMF18 beinhaltet acht asynchrone einzeln wählbare V.24-Schnittstellen (maximal 19,2 KBaud) oder IFSS-Schnittstellen (maximal 9,6 KBaud, 500 m), eine synchrone V.24-Schnittstelle (maximal 19,2 KBaud) sowie einen Parallel-E/A-Kanal in den Varianten IFSP-A-Kanal und Centronics oder AMF18-Parallelinterface und ZP18-Druckerinterface.

Weitere Peripheriegeräte von Robotron für den K 1840 sind der **Plotter K 6411** für A2-Format mit einer Geschwindigkeit von 600 mm/s sowie das **Digitalisiergerät K 6404.20** für A0-Format mit einer Genauigkeit von ±0,1 mm (Meßwertaufnehmer: Lupe) bzw. ±0,5 mm (Meßwertaufnehmer: Stift).

Die von Robotron angebotenen Drukker sind der Robotron-Typenraddrukker 1152/257, der Robotron-Nadeldrucker K 6313/K 6314 sowie Videotron-Zeilendrucker der Serien VT 27000 und VT 23000 (siehe Tafel).

Der **Seriendrucker 1152/Modell 257** ist ein Typenraddrucker mit IFSS-Interface und einer Druckgeschwindigkeit von 45 Zeichen pro Sekunde. Der **Nadeldrucker K 6313** bzw. **K 6314** kann mit 94 alphanumerischen Zeichen oder im Grafikmode drucken. Er hat ebenfalls eine IFSS-Schnittstelle und eine Druckgeschwindigkeit von 100 Zeichen pro Sekunde. Die **Paralleldrucker VT 27000/VT 23000** arbeiten mit einer Geschwindigkeit von 300 bis 900 Zeilen pro Minute und besitzen eine IFSP-Schnittstelle. Der VT 27065 kann mit einem kyrillischen Zeichensatz drucken.

Der K 1840 ist selbstverständlich auch netzfähig. Mit dem **Lokalnetzcontroller K 8681** (RONAS) kann über den **Transceiver K 8602** ein Anschluß an ROLANET 2 hergestellt werden. Das ethernet-kompatible ROLANET 2 gewährleistet eine Übertragungsrate von 10 MBit/s, bei Verwendung von Koaxialkabel jedoch nur eine Entfernung von 500 m (dieser Mangel sollte mit dem Einsatz von Lichtleitern überwunden werden können).

Eine Rechnerkopplung ist weiterhin über die **Datennahübertragungseinrichtung** (DNÜ) **K 8172** möglich. Sie erlaubt mit 2-Draht-Leitungen eine Entfernung von 2 km (bei 19,2 KBaud) bis 20 km (bei 1,2 KBaud). Mit 4-Draht-Leitungen sind Entfernungen von 5 km (bei 19,2 KBaud) bis 30 km (bei 1,2 KBaud) möglich. Die Datenübertragung kann mit der DNÜ asynchron und synchron erfolgen.

Weiterhin sind zwei Modemtypen verfügbar. Der **Modem VM 2400** vom VEB Nachrichtenelektronik Leipzig läßt synchronen Zwei- bzw. Vierdrahtbetrieb mit 1200 oder 2400 Baud zu. Von Telefongyar (Ungarn) kann der **Modem TAM 1200** bereitgestellt werden. Er kann ebenfalls über zwei- oder Vierdrahtleitung kommunizieren. Seine Übertragungsgeschwindigkeiten betragen 600 bzw. 1200 Baud, die Übertragung kann asynchron oder synchron erfolgen.

Die beiden Betriebssysteme SVP 1800 und MUTOS 1800 ermöglichen den Mehrnutzerbetrieb (maximal 16 Nutzer) einschließlich des Schutzes vor gegenseitigen Störungen.

Was ist ein Cache?

Das Wort Cache (sprich: käsch) kann man aus dem Englischen mit Versteckspeicher übersetzen. Es bezeichnet einen Hochgeschwindigkeitsspeicher, der ursprünglich in Großrechnern, aber zunehmend auch in Mikrorechnern die Wartezeiten der zentralen Verarbeitungseinheit (CPU) reduziert. Der Cache wird wiederum von einem Cache-Controller gesteuert. Warum ist das nötig?

Die Geschwindigkeit kostengünstiger RAM-Schaltkreise ist geringer als die der CPUs. Deshalb zwingen die RAMs bei hohen Rechnertaktraten die CPU ständig zum Warten. Die CPU muß Wartezyklen (Wait-States) einschieben oder mit einer geringeren Taktrate arbeiten. Oftmals sind Rechner, die mit einer geringeren Taktrate, aber ohne Wait-States arbeiten, genauso schnell wie die mit der größeren Taktrate. Nun läßt man einfach den Cache *vorausdenken.* Er wird beispielsweise mit den Befehlen oder Daten, die der Rechner als nächste *(wahrscheinlich)* benötigt, geladen. Für das Holen der Informationen aus dem Cache braucht die CPU auch bei höchsten Taktraten keine Wait-States. Das klappt natürlich nur, wenn der Rechner nicht plötzlich eine Sprungoperation ausführt oder wegen eines Interrupts das Holen von Daten beendet. (Die Zahl der CPU-Zugriffe, bei denen es geklappt hat, nennt man Hitrate). Um kostengünstig zu bleiben, muß der Cache gegenüber dem Hauptspeicher natürlich klein sein.

Der Deskpro 386/25 von Compaq beispielsweise beinhaltet bei maximal 16 MByte RAM (auf der Systemplatine) einen Cache von 32 KByte mit 25 ns Zugriffszeit. Auch IBM hat sich jetzt mit dem PS/2 Modell 70 dem Trend des Cache-Einsatzes in Mikrorechnern angeschlossen. Das Modell 70 verfügt bei maximal 6 MByte RAM (auf der Systemplatine) über einen 64-KByte-Cache mit 40 ns Zugriffszeit. Beide Rechner erreichen mit dem Cache-Controller 82385 von Intel eine Hitrate von 95%. *MP*

Das Betriebssystem **SVP 1800** läßt Stapel- und Echtzeitverarbeitung zu. Es kann in vier hierarchischen Betriebssystemmodi (Kernel, Executive, Supervisor, User) arbeiten. Bezüglich der Befehlsausführung werden zwei Modi unterschieden, der RVS-Modus, in dem alle SVP-1800-Programme (mit 32 Bit Breite) ausgeführt werden, sowie der Kompatibilitätsmodus, in dem der Grundbefehlssatz der 16-Bit-Rechner K 1600, CM3, CM4 realisiert wird. Für CAD/CAM-Anwendungen steht im SVP 1800 das **Grafische Kernsystem** (GKS) **1800** zur Verfügung. Außerdem können drei Geometriemodellierer genutzt werden:

- Modellierungssystem Geometrische Konstruktion **GEKO 1800**
- Geometriemodellierungssystem **GEMO 1800**
- Geometriebausteinsystem **GBS 1800.**

Für Datenbankarbeit bietet Robotron das relationale Datenbanksystem **ALLDBS** und für Rechnerkopplung und die Arbeit im ROLANET 2 das Softwarepaket SKRNET an. Weiterhin hat Robotron das UNIX-kompatible **MUTOS 1800** implementiert, das besonders für Programmentwicklung und Dokumentationserstellung (jedoch nicht für Echtzeitaufgaben) geeignet ist. Damit verfolgt Robotron die Strategie, für jeden neuen Rechnertyp ein UNIX-kompatibles Betriebssystem (als Zweitbetriebssystem) anzubieten (siehe auch MP 8/88). Als Datenbanksystem kann das System **DABA32**, das wie das System ALLDBS Mehrnutzerbetrieb erlaubt, verwendet werden. Als Netzsoftware ist unter MUTOS 1800 nur das Programmpaket **UUCP** für die Kopplung über IFSS/V.24, DNÜ oder Modem verfügbar. Die Nutzung von ROLANET 2 wird (noch) nicht unterstützt. Für beide Betriebssysteme stehen zur Zeit Compiler für **C**, **MODULA-2** und **FORTRAN 77** zur Verfügung.

MP-HK

Technische Daten

Operationsgeschwindigkeit	1,1 MIPS
als Terminal anschließbar	K 8911.80, K 8918, PC 1715, AC 7100/7150, EC 1834
Nutzeranzahl	16
virtueller Adreßraum	4 GByte
Steckplätze	44
Standardbestückung	3 AMF18 mit je 8 asynchronen IFSS-/V.24-Interfaces, mit je 1 synchronen V.24-Schnittstelle und je 1 Parallelschnittstelle (IFSP-A, Centronics oder AMF18. ZP18)

Prozessor

Typ	2/3-Adreßmaschine, mikroprogrammiert
Mikroprogrammspeicher	8 KWorte je 99 Bit, bestehend aus: 4 KWorte Festwertspeicher, 4 KWorte Schreib-/Lesespeicher
Interne Datenbreite	32 Bit
Cache	8 KByte, 290 ns mittlere Zugriffszeit
Allgemeine Register	16 × 32 Bit
Interruptebenen	32
Zeitgebereinrichtung	Uhr für Tageszeit und Datum sowie Intervallzeitgeber
Datentypen	– Integer – Gleitkomma – numerische und alphanumerische Zeichenketten – gepackte binär verschlüsselte Dezimalzahlen – Bitfelder variabler Länge
Adressierungsarten	9
Interner Synchronbus	13,3 MBytes/s, 200 ns Zykluszeit
Spezielle Einrichtungen	– Gleitkommabeschleuniger und -zusatz, Nutzersteuerspeicher
Standardmerkmale	– Netzausfallüberwachung – automatischer Restart

Hauptspeicher

Kapazität	8 MByte Standardausrüstung, aufrüstbar bis 16 MByte, in Schritten von 2 MByte
Schaltkreisbasis	64-KBit-NMOS

Peripherie

Festplattenspeicher robotron K 5502

Kapazität	124 MByte
Übertragungsgeschwindigkeit	806 KByte/s
Zugriffszeit	52,5 ms
maximale Anzahl	8

Wechselplattenspeicher	CM 5416	CM 5404
Kapazität	174 MByte	88 MByte
Übertragungsgeschwindigkeit	806 KByte/s	
Zugriffszeit	30 ms	
maximale Anzahl	8	

Magnetbandspeicher	CM 5306	CM 5309	CM 5308
Bandgeschwindigkeit	1,9 m/s	1,14 m/s	0,63 m/s
Datenrate, KByte	60/120	36/72	20/40
Kapazität	40 MByte	40 MByte	12,5 MByte

Typenraddrucker robotron 1152/Modell 257

Druckgeschwindigkeit	45 Zeichen/s
Zeichen/Zeile	132, 158, 197

Nadeldrucker robotron K 6313/K 6314

Druckgeschwindigkeit	100 Zeichen/s
Zeichen/Zeile	K 6313: 80, 96, 132; K 6314: 132, 165, 198

Zeilendrucker

	Zeilen/min	Zeichen/Zeile
VT 27060	600	136
VT 27065	660	136
VT 27090	900	136
VT 23300	300	132
VT 23600	600	132

Flachbettplotter robotron K 6411

Arbeitsfläche	A2
Anzahl der Werkzeuge	8
Zeichengeschwindigkeit	600 mm/s

Digitalisiergerät robotron K 6404.20

Arbeitsfläche	A0
Meßprinzip	induktiv
Meßwertaufnehmer	Stift/Lupe

Datenübertragungseinrichtung robotron K 8172

Prozedur	asynchron/synchron
Übertragungsgeschwindigkeit bei Vierdraht-Übertragung	19,2 KBaud bis 5 km; 1200 Baud bis 30 km

Modem VM 2400

Prozedur	synchron
Übertragungsgeschwindigkeit	1200/2400 Baud

Modem TAM 1200

Prozedur	asynchron/synchron
Übertragungsgeschwindigkeit	600/1200 Baud

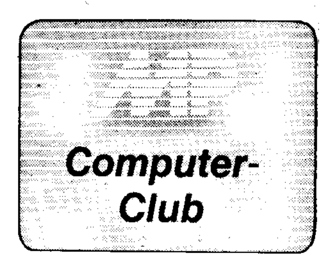
Fortsetzung von S. 310

Literatur

/1/ Conrad, M.: Nachnutzbare K-1520-kompatible OEM-Ergänzungsbaugruppen. Radio, Ferns., Elektron. 36 (1987) 3, S. 185
/2/ Erweiterungsbaugruppen für das System K 1520. Radio, Ferns., Elektron. 32 (1983) 11, S. 683
/3/ Weitere K-1520-Erweiterungsbaugruppen aus dem Applikationszentrum Mikroelektronik Erfurt. Radio, Ferns., Elektron. 34 (1985) 2, S. 67
/4/ Klingohr, B.: Koppelmodul für K 1520. Radio, Ferns., Elektron. 36 (1987) 5, S. 321
/5/ Quednow, W.: Videosteuerung VIS 3 mit GDC U 82720 D. Mikroprozessortechnik 2 (1988) 3, S. 66
/6/ Büttner, P.: In-Circuit-Echtzeitemulator für U 880 D. Radio, Ferns., Elektron. 36 (1987) 2, S. 87
/7/ Kämpf, W.: Einkartenrechner mit Einchip-Mikrorechner U 884 M. Radio, Ferns., Elektron. 35 (1986) 11, S. 690
/8/ Experimentierleiterplatte für die EMR U 882/884. Radio, Ferns., Elektron. 36 (1987) 8, S. 475
/9/ Waldmann, J.; Hoffmann, T.; Janke, E.: Alphanumerische Zeichendarstellung mit der LED-Anzeige VQC 10. Radio, Ferns., Elektron. 37 (1988) 2. S. 87
/10/ MRES-Software für Programmiermodul EPROM 1. Radio, Ferns., Elektron. 36 (1987) 8, S. 475
/11/ Programmiermodul für PC 1715. Radio, Ferns., Elektron. 36 (1987) 1, S. 4
/12/ Wrenzitzki, J.: EPROM-Programmiereinrichtungen aus dem IfAM-Erfurt. Mikroprozessortechnik 2 (1988) 11
/13/ Kämpf, W.: 16-Kanal-Analogeingabe für den Mikrorechner K 1520. Applikative Information 7 (1986) 4

✉ KONTAKT ☎

VEB Mikroelektronik „Karl Marx" Erfurt, IfAM Erfurt (Cl), Rudolfstraße 47, Erfurt, 5020, Tel.: 6 21 02/18 (NN-Lp) oder 6 21 02/36 (Lp-Entwurf und -Fertigung)



Taschenrechner-Programmbaustein in BASIC

Oftmals erweist es sich als notwendig, daß zur Überprüfung von angezeigten Werten oder zur Bestimmung von im Dialog geforderten Eingabewerten ein Taschenrechner zu Hilfe genommen werden muß. Das erscheint paradox, aber um das laufende Programm nicht abbrechen zu müssen, ist es in den meisten Fällen unumgänglich. Das hier vorgestellte kleine BASIC-Programm enthält ab Zeile 10 000 einen Modul, der den Taschenrechner in den 4 Grundrechenarten einschließlich Klammern ersetzen soll.
Als Zahlenbereich gilt für dieses Programm der Bereich der reellen Zahlen, ausgenommen davon sind Zahlen in Festkommadarstellung.
Erarbeitet wurde die Programmversion für den PC 1715 unter der Einschränkung, daß nur Standard-BASIC zum Einsatz kommt und als Anwendersteuerzeichen nur die 08 (zum Cursortransport nach links) verwendet wird. Dadurch müßte eine Implementierung auf anderen Computern möglich sein.

Funktionsbeschreibung

Der Programmbeginn liegt mit der Zeilennummer 10 000 im allgemeinen weit genug außerhalb des Programmes, von dem aus der Taschenrechner-Modul genutzt werden soll und ermöglicht damit eine Nachladbarkeit an jedes Programm, soweit es die Zeilennummer 9999 nicht überschreitet, der Arbeitsspeicher den notwendigen Platz noch frei hat und die genutzten Variablennamen nicht anderweitig Verwendung finden.
Der zu berechnende Ausdruck ist entsprechend den mathematischen Regeln einzugeben (ohne Gleichheitszeichen am Ende), mit RETURN abzuschließen und wird in der Zeichenkettenvariablen F$ abgelegt. Als erstes erfolgt ein Test, ob nur zulässige Zeichen und die gleiche Anzahl von „Klammer auf" wie „Klammer zu" vorliegen.
Im Falle der Erkennung eines solchen Fehlers wird eine entsprechende Fehlermitteilung ausgegeben.
Verläuft der Test ohne Auffinden eines Fehlers, so kann die Berechnung beginnen.
Zuerst werden alle Klammerausdrücke bereinigt; begonnen wird mit dem Klammerausdruck, der ein erstes „Klammer zu" aufzuweisen hat. Das erste davor liegende „Klammer auf" liefert dann die zweite Begrenzung des Klammerausdruckes, welcher ohne Klammern in der Zeichenkettenvariablen A$ abgelegt wird.
Die Zeichenkettenvariable A$ wird dann in der Routine ab 10 130 ausgewertet und entsprechend den mathematischen Regeln in 2 numerische Werte zerlegt und berechnet. Das Ergebnis der Rechnung wird dann in der Zeichenkettenvariablen ER$ abgelegt.
Im Anschluß daran erfolgt der Test, ob ER$ eine Zahl in Festkommadarstellung ist, und es wird gesichert, daß das Ergebnis in jedem Falle in Gleitkommadarstellung vorliegt.
Das Vorzeichen von ER$ ist entweder „+" oder „−".
Nach der Berechnung und einer eventuellen Umwandlung wird ER$ anstelle des vorher entnommenen Ausdruckes in die Zeichenkettenvariable A$ zurückgeschrieben.
Dieser Vorgang wiederholt sich, bis der klammerlose Ausdruck nur noch einen Operanden enthält. Dieser Operand wird in den ursprünglichen Ausdruck F$ in den Bereich eingeordnet, aus dem zuvor der klammerlose Ausdruck entnommen wurde.
Danach setzt sich die Suche nach weiteren Klammerausdrücken fort. Sind keine Klammerausdrücke mehr in F$ enthalten, so wird nochmals die Routine ab 10 130 genutzt, um das endgültige Ergebnis zu ermitteln und auszugeben.

Anwendungsbeispiel zum Taschenrechnermodul

Aufgabe soll es sein, die Eingabe eines numerischen Wertes zu programmieren. Die Eingabe wird in der Dimension „Minuten" verlangt, dem Nutzer liegt aber eine Angabe in Minuten oder Stunden oder Tagen usw. vor.
Um dem Nutzer den denkbaren Griff zum Taschenrechner zu ersparen, soll ihm durch Betätigen der Taste „T" der Taschenrechnermodul in jedem Moment der Eingabe zur Verfügung stehen. Nach Ergebnisausgabe soll durch Betätigen einer beliebigen Taste die Eingabe fortgesetzt werden können. Dabei soll die kleine Nebenrechnung wieder gelöscht werden.
Eine Lösungsvariante dazu ist im Programm-Listing dargestellt, der eigentliche Taschenrechner-Modul beginnt ab Zeile 10 000. Die Lösung kann ebenfalls als Modul für jede beliebige Eingabe genutzt werden, indem auf Zeile 170 ein RETURN untergebracht wird. Zum Beispiel: Eingabe eines weiteren Wertes W

200 PRINT"WERT:";:ZA$="":
GOSUB 30:W=VAL (ZA$)

Um einen noch größeren Effekt bei der Nutzung dieses Taschenrechner-Moduls zu erreichen, wäre es denkbar, auch die wissenschaftlichen Funktionen nutzen zu können. Der Programmumfang vergrößert sich dadurch aber erheblich, und die Abarbeitungszeit steigt.
Als Lösungsstrategie wäre aber das gleiche Prinzip denkbar. Für Hinweise zu besseren Lösungsstrategien, die zu einer schnelleren Abarbeitung führen, wäre der Autor sehr dankbar.

Michael Schneider

```
10      ZA$=""
20      PRINT"ZEIT (MINUTEN): ";
30      T$=INKEY$:IF T$="" THEN GOTO 30
40      IF T$="T" THEN GOSUB 10000 ELSE GOTO 80
50      T$=INKEY$:IF T$="" THEN GOTO 50
60      AN=LEN(EG$)+1+LEN(F$)+T:FOR  N=1  TO  AN:PRINT  CHR$(8)+
" "+CHR$(8);:NEXT N
70      GOTO 30
80      IF ASC(T$)>47 AND ASC(T$)<58 THEN ZA$=ZA$+T$:PRINT T$;:
GOTO 30
90      IF T$="." THEN ZA$=ZA$+T$:PRINT T$;:GOTO 30
100     IF T$="-" THEN ZA$=ZA$+T$:PRINT T$;:GOTO 30
110     IF ASC(T$)=8 THEN GOTO 120 ELSE GOTO 140
120     IF LEN(ZA$)>0 THEN PRINT CHR$(8);:ZA$=LEFT$(ZA$,LEN(ZA$)-1)
130     GOTO 30
140     IF ASC(T$)=13 THEN GOTO 160
150     GOTO 30
160     PRINT:ZA=VAL(ZA$)
170     END
10000   REM PROGRAMM FORMI ALS TASCHENRECHNER (FORMELINTERPRETER)
10010   INPUT;">",F$:F1$=F$
10020   GOTO 10590
10030   REM IN F$ STECKT DIE FORMEL
10040   Z=INSTR(F$,")")
10050   IF Z=0 THEN GOTO 10530
10060   Z1=Z
10070   IF MID$(F$,Z1,1)="(" THEN GOTO 10100
10080   Z1=Z1-1
10090   GOTO 10070
10100   A$=MID$(F$,Z1+1,Z-Z1-1)
10110   GOSUB 10130
10120   GOTO 10500
10130   REM BERECHNEN DER KETTE OHNE KLAMMERN
10140   OP$="/"
10150   MA=INSTR(A$,OP$)
10160   IF MA<2 THEN GOTO 10450
10170   LI=MA
10180   LI=LI-1:IF LI<1 THEN GOTO 10220
10190   IF  ASC(MID$(A$,LI,1))>47 AND  ASC(MID$(A$,LI,1))<58  THEN
GOTO 10180
10200   IF MID$(A$,LI,1)="." THEN GOTO 10180
10210   IF MID$(A$,LI,1)="-" OR MID$(A$,LI,1)="+" THEN LI=LI-1
10220   LO$=MID$(A$,LI+1,MA-LI-1)
10230   RE=MA+1
10240   RE=RE+1:IF RE>LEN(A$) THEN GOTO 10270
10250   IF  ASC(MID$(A$,RE,1))>47 AND  ASC(MID$(A$,RE,1))<58  THEN
GOTO 10240
10260   IF MID$(A$,RE,1)="." THEN GOTO 10240
10270   RO$=MID$(A$,MA+1,RE-MA-1)
10280   IF OP$="*" THEN EG$=STR$(VAL(LO$)*VAL(RO$))
10290   IF OP$="/" THEN EG$=STR$(VAL(LO$)/VAL(RO$))
10300   IF OP$="+" THEN EG$=STR$(VAL(LO$)+VAL(RO$))
10310   IF OP$="-" THEN EG$=STR$(VAL(LO$)-VAL(RO$))
10320   IF LEFT$(EG$,1)=" " THEN EG$=RIGHT$(EG$,LEN(EG$)-1):
GOTO 10320
10330   IF LEFT$(EG$,1)="-" THEN VZ$="-" ELSE VZ$=""
10340   GOTO 10360
10350   IF LEFT$(EG$,1)<>"-" AND LI>0 THEN EG$="+"+EG$:GOTO  10420
ELSE GOTO 10420
10360   IF LEN(EG$)<6 THEN GOTO 10350
10370   IF MID$(EG$,LEN(EG$)-3,1)="D" OR MID$(EG$,LEN(EG$)-3,1)="E"
THEN GOTO 10380 ELSE GOTO 10350
10380   AN=VAL(MID$(EG$,LEN(EG$)-1,2))
10390   IF MID$(EG$,LEN(EG$)-2,1)="-" THEN GOTO 10410
10400   EG$=VZ$+LEFT$(EG$,1)+MID$(EG$,3,LEN(EG$)-6)+
STRING$(AN-LEN(EG$)+6,"0"):GOTO 10350
10410   EG$=VZ$+"0."+STRING$(AN-1,"0")+LEFT$(EG$,1)+
MID$(EG$,3,LEN(EG$)-6):GOTO 10350
10420   A$=LEFT$(A$,LI)+EG$+RIGHT$(A$,LEN(A$)-RE+1)
10430   REM PRINT LO$,RO$,OP$,EG$,A$
10440   GOTO 10150
10450   IF OP$="/" THEN OP$="*":GOTO 10150
10460   IF OP$="*" THEN OP$="-":GOTO 10150
10470   IF OP$="-" THEN OP$="+":GOTO 10150
10480   REM IN A$ STEHT EINE ZEICHENKETTE OHNE OPERATOREN
10490   RETURN
10500   REM SUBSTITUTION DES KLAMMERAUSDRUCKES DURCH ERGEBNIS A$
10510   F$=LEFT$(F$,Z1-1)+A$+RIGHT$(F$,LEN(F$)-Z)
10520   GOTO 10040
10530   REM ES SIND KEINE KLAMMERN MEHR DA
10540   A$=F$
10550   GOSUB 10130
10560   IF INSTR(EG$,".")>0 AND LEN(EG$)>16 THEN EG$=LEFT$(EG$,16)
10570   PRINT "="+EG$;:T=1
10580   GOTO 10740
10590   REM TEST AUF GUELTIGE ZEICHEN
10600   KA=0:KZ=0:FA=0:FE=0
10610   FOR N=1 TO LEN(F$)
10620   A$=MID$(F$,N,1)
10630   IF A$="(" THEN KA=KA+1
10640   IF A$=")" THEN KZ=KZ+1
10650   IF ASC(A$)<48 OR ASC(A$)>57 THEN FA=FA+1
10660   IF  A$="."  OR A$="*" OR A$="/" OR A$="+"  OR  A$="("  OR
A$=")" OR A$="-" THEN FA=FA-1
10670   NEXT N
10680   IF KA<KZ THEN T$="WENIGER":FE=1:T=7
10690   IF KA>KZ THEN T$="MEHR":FE=1:T=4
10700   IF FA>0 THEN T$=STR$(FA)+"UNZUL.ZEICHEN":
FE=2:T=LEN(STR$(FA))+15
10710   IF FE=1 THEN PRINT T$+" KLAMMERN AUF ALS ZU";:T=T+20
10720   IF FE=2 THEN PRINT T$;
10730   IF FE=0 THEN T=0:GOTO 10030
10740   RETURN
```

Zeichensatzänderung beim KC 85/3

Im Anleitungsmaterial zum KC 85/3 ist eine Zeichensatzerweiterung beschrieben. Es können dort selbst Zeichen definiert werden, die aber nur mit der CHR$-Funktion aufgerufen werden. Nun interessierte uns die Möglichkeit, die Tastatur des Computers mit anderen Zeichen zu belegen, indem der vorhandene Zeichensatz verändert wird.
Für diesen Zweck stellt das folgende Programm ein Hilfsmittel dar. Die Zeichentabellen des KC 85/3 beginnen bei EE00H und FE00H im ROM und können dort nicht verändert werden. Deshalb werden diese beiden Tabellen in den Zeilen 20 bis 50 in die Bereiche ab BA00H und BC00H kopiert. Dies dauert etwa 30 Sekunden.
Für Großbuchstaben, Ziffern und Sonderzeichen sind die Zeiger B7A6H und B7A7H zuständig. Der Inhalt von B7A6H beträgt bereits 00H, so daß nur noch B7A7H auf BAH (186) gestellt werden muß. Dies erfolgt in Zeile 60. Analog dazu wird für die 2. Zeichentabelle die Umstellung für B7A9H in Zeile 70 vorgenommen. Nach der Anzeige des Menüs hat man die Möglichkeit, über F1 und nach entsprechendem Tastendruck einzelne Zeichen aufzurufen. Dabei erscheint das Zeichen groß mit Speicherstellen und Bytewerten auf dem Bildschirm (Zeilen 220 bis 320).
Durch Betätigen von F2 kommt man in den Editiermodus. Nun kann das Zeichen mit den Cursortasten und F5, F6 verändert werden, wobei das gerade bearbeitete Bit rot blinkt. Im Programm sind die Zeilen 330 und 500 dafür zuständig.
Nach der Änderung des Zeichens wird es mit F3 abgespeichert. Es erscheinen dann die Bytewerte und die Speicherstellen des neuen Zeichens. Diese Werte können in anderen Programmen für DATA- und VPOKE- (oder POKE-) Befehle genutzt werden. Man kann also auch beim Einstellen von einzelnen Zeichen, die über die CHR$-Funktion aufgerufen werden sollen, das Programm nutzen. Diese Errechnung erfolgt in den Zeilen 510 bis 610. Sind viele Zeichen geändert worden, besteht die Möglichkeit, nach Betätigung von F4 den gesamten Zeichensatz abzuspeichern. Dazu schaltet das Programm ins Hauptmenü um, und der Bereich BA00H bis BE00H kann aufgezeichnet werden. Dieser neue Zeichensatz kann später mit LOAD oder BLOAD genutzt werden. Sollte er nicht hintereinander verändert werden, so muß nach dem Einladen des unvollständig geänderten Zeichensatzes das Programm mit GOTO 60 gestartet werden. Eine andere Möglichkeit ist, nach der Zeichenänderung mit BRK und NEW zu arbeiten und dann einen DATA-Zeilenprogrammgenerator einzuladen.
Ein solches Programm ist in MP 10/1987, Seite 318, von K.-D. Kirves beschrieben worden. Man muß bei der späteren Nutzung des neuen Zeichensatzes nur an die Umstellung der Zeiger denken.
Mit Hilfe des Programms ist man in der Lage, die unterschiedlichsten Zeichen (kyrillische Buchstaben, Schaltzeichen u. a.) selbst zu definieren und einzusetzen. Es ist aber zu beachten, daß z. B. eine Neubelegung der Tastenfolge *PRINT mit anderen Zeichen* trotzdem noch den PRINT-Befehl beinhaltet. Weiterhin kann man sehr gut die Entstehung der einzelnen Zeichen demonstrieren.
Bestimmt ergibt sich hier für AG's und Computerclubs ein weites Experimentierfeld.

Hans-Jürgen Eicke

```
10 CLS:PRINTAT(15,3);"BITTE WARTEN : ZEICHENTABELLEN WERDEN KOPIERT:30 SEK"
20 FORX=0TO511
30   VPOKE(14848+X),PEEK(-4608+X):REM 1.ZEICHENTABELLE KOPIEREN
40   VPOKE(15360+X),PEEK(-512+X):REM 2.ZEICHENTABELLE KOPIEREN
50   NEXTX
60 VPOKE14247,186:REM ZEIGER B7A7 GROSSBUCHSTABEN UMSTELLEN
70 VPOKE14249,188:REM ZEIGER B7A9 KLEINBUCHSTABEN UMSTELLEN
80 REM MENUANZEIGE
90 CLS:WINDOW0,16,0,39:INK7:PRINTAT(19,5);"F1 : NEUES ZEICHEN"
100 PRINTAT(20,5);"F2 : ZEICHEN AENDERN"
110 PRINTAT(21,5);"F3 : GEAENDERTES ZEICHEN SPEICHERN"
120 PRINTAT(22,5);"F4 : ZEICHENSATZ ABSPEICHERN/ENDE"
130 PRINTAT(23,5);"F5 : PUNKT SETZEN":PRINTAT(24,5);"F6 : PUNKT LOESCHEN"
140 PRINTAT(27,5);COLOR18,6;"BITTE AUSWAEHLEN"
150 A=PEEK(509):IF A<>241ANDA<>242ANDA<>244THEN150
160 PRINTAT(27,5);COLOR7,1;"                    "
170 IFA=244THEN620
180 IFA=242THEN330
190 CLS:PRINTAT(8,2);"BITTE TASTE MIT GEWUENSCHTEM ZEICHEN    DRUECKEN"
200 B=PEEK(509):IFB<32ORB>127THEN200
210 IFB=>32ANDB<=95THENSA=14848:A1=32:ELSESA=15624:A1=97
220 REM 1.SPEICHERSTELLE DES ZEICHENS SUCHEN
230 Q=B-A1:CLS:INK7:SP=SA+8*Q:PRINTAT(4,7);"BIT 76543210"
240 REM BYTES AUS 8 SPEICHERSTELLEN LESEN
250   FORY=0TO7
260   SP(Y)=SP+Y:Z=VPEEK(SP(Y)):Z(Y)=Z:LOCATEY+5,11
270    FORX=7TO0STEP-1
280    REM BYTES IN BIT7 BIS BIT0 ZERLEGEN
290    IFZ-2^X<0THENA$(Y,X)=CHR$(46):ELSEA$(Y,X)=CHR$(91):Z=Z-2^X
300    PRINTA$(Y,X);:IFX=0THENPRINT"    ";Z(Y);"    ";SP(Y);"    "
310    NEXTX:NEXTY
320 PRINTAT(10,5);CHR$(B):GOTO140
330 X=7:Y=0:GOSUB690
340 A=PEEK(509)
350 IFA<>8ANDA<>9ANDA<>10ANDA<>11ANDA<>243ANDA<>245ANDA<>246THEN340
360 IFA>11THEN460:ELSEA=A-7
370 ONAGOTO380,400,420,440
380 GOSUB680:IFX=7THENX=0:ELSEX=X+1:REM BEWEGUNG CURSOR LINKS
390 GOSUB690:GOTO340
400 GOSUB680:IFX=0THENX=7:ELSEX=X-1:REM BEWEGUNG CURSOR RECHTS
410 GOSUB690:GOTO340
420 GOSUB680:IFY=7THENY=0:ELSEY=Y+1:REM BEWEGUNG CURSOR HOCH
430 GOSUB690:GOTO340
440 GOSUB680:IFY=0THENY=7:ELSEY=Y-1:REM BEWEGUNG CURSOR RUNTER
450 GOSUB690:GOTO340
460 IFA=243THEN510
470 IFA=245THENA$(Y,X)=CHR$(91):ELSEA$(Y,X)=CHR$(46):REM PUNKTE SETZEN
480 GOSUB680:IFX=0ANDY=7THENX=7:Y=0:GOTO500
490 IFX=0THENX=7:Y=Y+1:ELSEX=X-1
500 GOSUB690:GOTO340
510   FORY=0TO7
520   Z(Y)=0
530    FORX=7TO0STEP-1:REM BITWERTE ABFRAGEN
540    IFA$(Y,X)=CHR$(91)THENZ(Y)=Z(Y)+2^X:REM BYTE BERECHNEN
550    IFX=0THENPRINTAT(Y+5,22);INK4;Z(Y);"    ";SP(Y);"   "
560    NEXTX
570   REM ERRECHNETES BYTE IN SPEICHERSTELLE SCHREIBEN
580   VPOKESP(Y),Z(Y):PRINTAT(10,5);" ":PRINTAT(10,5);CHR$(B)
590   NEXTY
600 GOTO140
610 REM ZEICHENTABELLE ABSAVEN
620 VPOKE14247,238:WINDOW
630 CLS:PRINT"NACH UMSCHALTUNG INS HAUPTMENU SPEICHER-BEREICH ABSAVEN"
640 PAUSE60:CLS
650 PRINTAT(1,0);"%SAVE BA00 BE00"
660 BYE
670 END
680 PRINTAT(Y+5,18-X);INK7;A$(Y,X):RETURN
690 PRINTAT(Y+5,18-X);INK18;A$(Y,X):RETURN
```

Z-1013-Service

Den Reparaturservice für den Mikrorechnerbausatz Z 1013 einschließlich Erweiterungsbaugruppen übernimmt ab sofort der
VEB Industrievertrieb Rundfunk und Fernsehen
BT Friedrichshain
Neue Bahnhofstr. 30
Berlin
1035
Annahme:
Montag–Freitag 10.00 Uhr–13.00 Uhr und
15.00 Uhr–19.00 Uhr oder Postversand

Büttner



Börse

MS-DOS-Implementation für nicht IBM-kompatible 16-Bit-Hardware

Nachdem für 16-Bit-Rechner umfangreiche leistungsfähige Software zur Verfügung steht, ergibt sich für Nutzer der 8-Bit-Technik die Frage, ob es möglich ist, eine Hardwareerweiterung für 8-Bit-Rechner aufzubauen, auf der MS-DOS und die darin verfügbare Software benutzt werden kann.

Die hardwaremäßigen Voraussetzungen für das Schaffen einer MS-DOS-Umgebung auf 8-Bit-Rechnern (A 51xx, PC 1715) sind lösbar durch eine Leiterkarte, die folgendermaßen ausgestattet ist:
- 8086/8088/V30-Prozessor
- ein frei verfügbarer RAM (beginnend mit Adresse 0) der Größe 256 bis 512 KByte; keinerlei Teile dieses RAMs werden durch Hardwarefunktionen benutzt.

Ferner wird an peripherer Hardware benötigt:
- eine beliebige Tastatur, die um so besser ist, je mehr unterscheidbare Tasten sie hat
- mindestens zwei Floppy-Laufwerke, Diskettenattribute (nach Möglichkeit) 80 Spuren, DS
- ein Bildschirm für 80*25 Zeichen und Attribute pro Zeichen. Ein 80*24-Bildschirm kann ebenfalls benutzt werden.
- eventuell ein Drucker.

Die Art des Anschlusses der peripheren Geräte ist prinzipiell ohne Bedeutung, er hat jedoch Einfluß auf die Gesamteffizienz des Systems.

Der Effektivitätszuwachs wird insbesondere erreicht für 16-Bit-Hardware ohne eigene Bildschirmbedienung. Wichtig ist dabei nur die Schnittstelle zur Bildschirmbedienung. Intelligente Zusatzhardware in Form eines 8-Bit-Rechners ist dafür viel zu langsam. Die Bereitstellung für den Bildschirm übernimmt vollständig der 16-Bit-Prozessor, der 8-Bit-Prozessor ist nur für den Transport zum Schirm zuständig. Auf der beschriebenen Hardware wurde MS-DOS implementiert. In dieser MS-DOS-Umgebung laufen u. a. TURBO-PASCAL 3.0, TURBO C, der Editor ne, TURBO BASIC. Mit dem Einschub dieser Leiterkarte und der Betriebssystemimplementation hat der 8-Bit-Rechner dieselbe Architektur wie ein A 7100, jedoch eine Geschwindigkeit, die dem XT entspricht und teilweise größer ist. Auf der gleichen Hardware laufen auch CP/M-86-Programme wie auf A 7100, z. B. REDABAS, TURBO-PASCAL und allgemeine Software.

Humboldt-Universität zu Berlin, Bereich Informationsverarbeitung, Unter den Linden 6, Berlin, 1086; Tel. 20 93 23 48

Krzikalla

Softwaremodule für den APR-1

Zur Erstellung von Software für den Arithmetikmodul APR-1 auf der Basis U 8032 mit Anpassung an die Systemstruktur des K 1520 für die 16-Bit-Festkommaoperationen wurde ein Assembler erarbeitet. Er ist unter UDOS lauffähig und realisiert die Assemblierung der vom VEB ZFT Mikroelektronik Dreden vorgeschlagenen Mnemoniks. Im Ergebnis liegt von den Inhalten der 5 zu programmierenden EPROMs des APR-1 jeweils ein Viertel vor. Der Assembler übersetzt Quellprogramme, die mit dem Texteditor unter UDOS 1526 erstellt wurden. Dabei sind bis zu 256 verschiedene Marken (bei maximal 256 Programmzeilen), Kommentare bzw. Kommentarzeilen und Absolutzuweisungen von 8- oder 16-Bit-Konstanten möglich. Die Befehlsfolgen für äußeres und inneres Steuerwerk können zur Assemblierung hintereinander in einem Quelltext stehen. Die Arbeit mit diesem Programmteil und dem Linker ist bildschirmunterstützt und braucht nicht weiter erläutert werden. Ein weiterer Vorteil für den Anwender liegt darin, daß neue oder veränderte Mnemoniks und Befehlsbytes in einer dafür vorgesehenen Datei des Assemblers verändert oder ergänzt werde können. Bei der Übersetzung von 2, 3 oder 4 verschiedenen Quellprogrammen erleichtert der Linker das Zusammenbinden der 10, 15 oder 20 Objektdateien auf 51 KByte lange Bereiche für das folgende EPROM-Programmieren.

Technische Universität Karl-Marx-Stadt, Sektion Automatisierungstechnik, PSF 964, Karl-Marx-Stadt, 9010

Schlegel/Dr. Ester

SCPX-Programm zum Austausch von Kassettendateien zwischen KC 85/x und PCs

Um diese Routine weitestgehend maschinenunabhängig zu machen, wird die Datenein-/-ausgabe über eine Standardschnittstelle durchgeführt. Die Routine wurde sowohl für eine parallele (PIO) als auch für eine serielle (V.24) Schnittstelle entwickelt. Ein direkt auf die Schnittstelle zu steckender Adapter stellt die Verbindung zum Tonbandgerät dar (Diodenbuchse). Die Stromversorgung des Adapters wird direkt durch die Schnittstelle realisiert, das heißt, es ist kein Eingriff in die Rechnerhardware notwendig.

Ziel der Programmentwicklung war es, eine Routine zu schaffen, die in der Lage ist, auch verstümmelte Daten noch lesen zu können. Die fehlerhaften Blöcke werden gekennzeichnet. Dadurch ist der Bediener in der Lage, beliebig viele Versuche zum Einlesen eines Fehlerblockes (z. B. sofort durch Bandrückspulen noch während des Lesevorganges) zu starten. Ein einmal als richtig erkannter Block wird bei wiederholtem Lesen nicht überschrieben.

Die Kassettenroutine legt im RAM einen 32 KByte großen Puffer für die Schreib-/Lesedaten an, dessen Lage im RAM wahlfrei ist. Im Puffer befindliche Daten lassen sich leicht auf Diskette schreiben und stehen somit zur Weiterbearbeitung am selben oder anderen Rechnern zur Verfügung. Zum Schreiben von Dateien für den KC muß ein spezieller Vorblock generiert werden, in dem die für den KC nötigen Parameter und Informationen enthalten sind. Die Erstellung dieses Vorblockes erfolgt mittels eines Anwendermenüs.

Die Routine arbeitet interruptgesteuert, die speziellen Portadressen lassen sich vom Anwender leicht einstellen (Generierungen für A 5120 und PC 1715 liegen vor). Das Programm ist eine wirksame Unterstützung für die Arbeit in Computerkabinetten, hat sich aber auch für die Erfassung von Daten an für den PC ungeeigneten Orten mittels KC bewährt.

Die angebotene Lösung beinhaltet:
- Schaltungsunterlagen für den seriellen und parallelen Adapter
- Software incl. Beschreibung.

Forschungszentrum für Mechanisierung und Energieanwendung in der Landwirtschaft der AdL der DDR, BfN, Gartenstr. 30, Schlieben, 7912; Tel. 5 33

Kneist

Speicherkarte (64 + 256) KByte für K 1520

Wir haben eine Speichersteckeinheit für den Mikrorechner **K 1520** entwickelt, die aus einem 64 KByte großen Hauptspeicher und einem 256 KByte großen Zusatzspeicher besteht. Konstruktiver Aufbau und Stromaufnahme (+5V, ca. 500 mA) gestatten einen änderungslosen Austausch mit der 64-KByte-Steckeinheit **K 3525** des Systems **K 1520**. Der 256 KByte große Zusatzspeicher kann bei entsprechender Software-Ergänzung sowohl als Erweiterung des ansteuerbaren RAM-Bereiches als auch in Form einer RAM-Disk eingesetzt werden.

Der Zusatzspeicher ist in 4 Seiten mit je 64 KByte aufgeteilt und arbeitet mit dem Hauptspeicher zusammen. Über jeweils einen der **U880**-Blockbefehle (**LDIR, LDDR, CPIR, CPDR, INIR, INDR, OTIR** bzw. **OTDR**) ist der Transfer von Datenmengen vom Hauptspeicher auf einen Bereich des Zusatzspeichers und umgekehrt bzw. zwischen beliebigen Speicherseiten und auch der direkte Transport von oder zu peripheren Baugruppen möglich. Mit Hilfe eines vorherigen **OUT**-Befehles (Portadresse wickelbar, voreingestellt auf FOH) werden Quelle und Ziel des Transfers festgelegt. Die Speicherkarte wird über eines der **K1520-MEMDI**-Signale gesperrt (Auswahl durch Wickelbrücken, voreingestellt auf **/MEMDI**).

Die Herstellung der (64 + 256)KByte-dRAM-Speicherkarte erfolgt auf der Basis des wissenschaftlichen Gerätebaues an der TU Karl-Marx-Stadt. Eine Auslieferung ist im Rahmen der Fertigungsmöglichkeiten ab 1989 gegeben. Zum Lieferumfang gehören die komplette Steckeinheit, die entsprechende Dokumentation (Stromlaufplan, Stückliste, Bestückungsplan und Kurzbeschreibung) und eine auf Nachnutzungsbasis mitgelieferte Software für den Einsatz des 256KByte-Speichers als **RAM**-Floppy (zusätzliches Laufwerk M). Diese Software für **CP/M**-kompatible Systeme wird auf User-Diskette in 2 Versionen bereitstehen:
- als nachladbares Programm (COM-Datei), das den TPA um ca. 2,5 KByte verringert und nach jedem Systemstart aufgerufen werden muß
- für den Systemprogrammierer als Einzelroutinen zur Floppy-Arbeit, die an den entsprechenden Stellen des BIOS eingebunden werden müssen (das setzt allerdings die Verfügbarkeit der BIOS-Quellen voraus, macht aber die RAM-Floppy-Arbeit sofort nach dem Systemstart möglich).

Technische Universität Karl-Marx-Stadt, Sektion Informationstechnik, Bereich Technik, Koll. Nestler, Reichenhainer Str. 70, Karl-Marx-Stadt, 9022; Tel. 5 61 32 52

Dr. Menzel

Generierung von TURBO-PASCAL-Quellprogrammen für die Bearbeitung von Dateien

Das von uns entwickelte Programm (CREATE) generiert nach Eingabe einer Dateistruktur innerhalb weniger Sekunden ein nach der Compilierung sofort lauffähiges TURBO-PASCAL-Quellprogramm zur Erzeugung und Bearbeitung der definierten Datei und einer Indexdatei.

Die von CREATE erzeugten Programme, die auf der Grundlage des Turbo-Access-Systems arbeiten, stellen Prozeduren zum Hinzufügen, Auffinden, Editieren, Löschen und Auflisten von Datensätzen sowie alle erforderlichen Menüs zur Verfügung. Der generierte Quelltext kann problemlos in ein größeres Turbo-Programmsystem eingebunden oder als eigenständiges Programm genutzt werden.

VWB Kombinat Spielwaren Sonneberg, Direktionsbereich Organisation und Datenverarbeitung, Oberlinder Str. 84, Sonneberg, 6412

H.-W. Großmann

Wir suchen

... eine Lösung zum Anschluß eines großen Bildschirmes bzw. Fernsehgerätes oder mehrerer kleiner Bildschirme an einen A 7100/A 7150, die für etwa 10 bis 15 Beratungsteilnehmer gute Sichtbedingungen ermöglicht.

VEB BMK Kohle und Energie, KBI Dresden, Strehler Str. 22, Dresden, 8020.

Pietzsch

... ein EPROM-Programmiergerät ab 2716 oder nachnutzbare Dokumentation.

Schnittstellen für Rechneranschluß IFSS oder V.24 (DEE) möglichst mit Software für A 7100.

VEB Elektro-Physikalische Werke Neuruppin, Erich-Dieckhoff-Straße, Neuruppin-Treskow, 1951; Tel. 6 14 56

Dr. Bernhardt



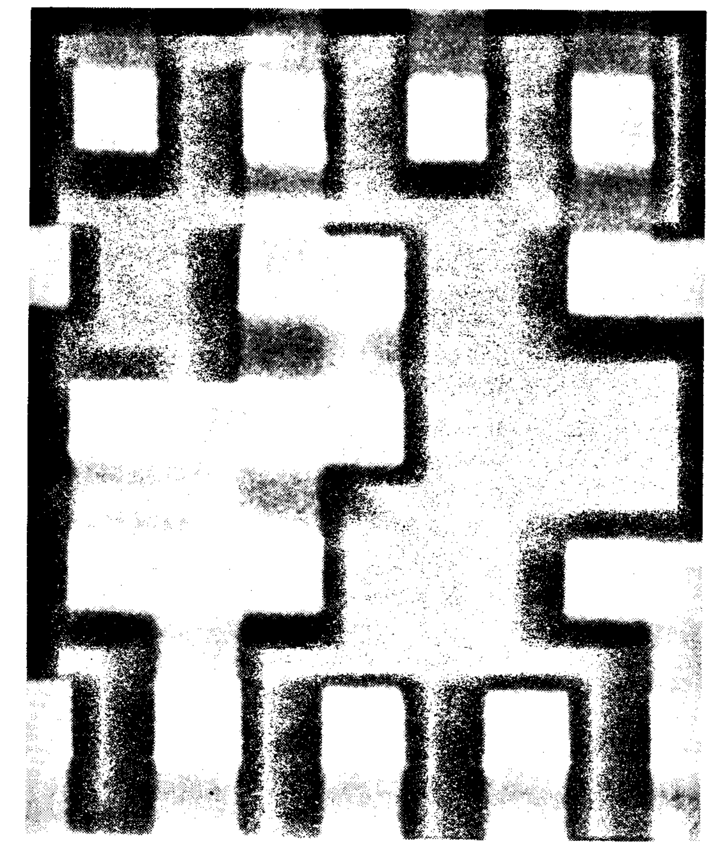
Entwicklungen und Tendenzen

Computer spricht Esperanto

Eine automatische Sprachsynthese für Esperanto wurde an der Technischen Universität Budapest erarbeitet. Das System ESPAROL kann jeden eingegebenen Esperanto-Text in richtiger Intonation aussprechen. Auch Namen und Sonderzeichen, soweit sie auf der Tastatur vorhanden sind, (=, +, % usw.) werden mitgesprochen.
Mit einem einfachen BASIC-Programm lassen sich relativ lange Texte mit Variablen sprechen. Die Aussprache kann verschieden schnell, flüsternd oder rauh usw. eingestellt werden. Selbst Lieder lassen sich programmieren.
Das System besteht aus einem Apparat, der mit dem Commodore 64 oder 128 verbunden wird und einem Programm auf Diskette. Für IBM-PC und kompatible Typen wurde eine spezielle Variante entwickelt. ESPAROL ist wahrscheinlich das erste Sprachsynthese-System für stark verbreitete Computer.
Es kann mit einem optischen Lesegerät, entwickelt im ungarischen Computerforschungszentrum SZKI, gekoppelt werden. So können sehschwache oder stark beschäftigte Personen sich Texte vom Computer vorlesen lassen.
Die mit dem Computer erstellten Texte können probegelesen werden. Die Schöpfer von ESPAROL, Koutny Ilona, Dr. Olaszy Gábor und Dr. Czapp László beschäftigen sich nun gemeinsam mit Studenten mit der Anwendung dieses Systems. Fertig sind bereits ein sprechendes Lehrprogramm für die Plansprache Esperanto und Spiele für den Unterricht.
Jeder Anwender kann seinen Wünschen entsprechend die Möglichkeiten des Systems durch einfache BASIC-Programme erweitern. Weitere Informationen können z. B. bei Koutny Ilona, Technische Universität Budapest, Stoczek u. 2; H-1111 erhalten werden.
aus FOKUSO 4/87 *Weber*

4-MBit-Chip bei der Arbeit beobachtet

Bei der Strukturfeinheit heutiger mikroelektronischer Schaltungen ist es nicht mehr möglich, elektrische Vorgänge auf dem Chip durch Meßspitzen zu erfassen. Bei Siemens wurde schon vor vielen Jahren begonnen, modifizierte Elektronenmikroskope für diese Untersuchungen zu verwenden. Man bedient sich dabei der Potentialkontrastmethode. Hierbei schlägt der Elektronenstrahl des Elektronenmikroskops aus der Oberfläche der Leiterbahnen Sekundärelektronen heraus. Je nach Potential der Leiterbahn ist die Anzahl der detektierten Sekundärelektronen verschieden: Je positiver die Spannung der Leiterbahn, desto weniger Elektronen verlassen die Oberfläche. Man erhält so ein Potentialbild der Oberfläche des Chips, an dem die positiven Leiterbahnen mit höherer Spannung dunkel und Leiterbahnen mit geringer Spannung heller erscheinen. Durch Takten des Elektronenstrahls relativ zur Arbeitsfrequenz des Speichers kann man den Speicher wie mit einem Stroboskop auch in seinem dynamischen Verhalten untersuchen.
Die bisher für dieses Verfahren eingesetzten Elektronenstrahlmeßgeräte sind im Prinzip modifizierte Rasterelektronenmikroskope, die eigentlich für Beschleunigungsspannungen von 20–40 kV ausgelegt sind und für die Elektronenstrahlmeßtechnik bei nur 1–3 kV betrieben werden. Zur Messung auf kleinen Strukturen im Sub-μm-Bereich, wie sie z. B. beim 4-MBit-Chip auftreten, reichen sie nicht mehr aus.
Siemens hat nun gemeinsam mit der ICT GmbH im Rahmen des 4-MBit-Projekts ein neues Elektronenstrahlmeßgerät entwickelt.
Kernstück ist eine elektronenoptische Säule, die speziell für niedrige Beschleunigungsspannungen zwischen 0,5 kV und 2,5 kV und hohen Sondenstrom (1–3 nA) ausgelegt wurde. Mit dem neuen Gerät kann auf Strukturen bis herab zu 0,5 μm gemessen werden (Leitbahnbreite = 5 × Ortsauflösung). Im stroboskopischen Betrieb ist eine Zeitauflösung bis herab zu 150 ps möglich. Durch die hohe Ortsauflösung – sie ist besser als bei manchen normalen Rasterelektronenmikroskopen – kann das Elektronenstrahlmeßgerät auch im Abbildungsbetrieb zum Erkennen prozeßbedingter Fehler eingesetzt werden.
Das Logikbild zeigt den Ausschnitt aus der Busstruktur eines 4-MBit-Speichers (Leitbahnbreite = Leitbahnabstand = 1,1 μm; dunkel – logisch 1, hell – logisch 0).
MP



Werkfoto

Neue Prozessoren für die 8086-Familie

Ein neuer 80286-kompatibler Prozessor kommt von der Nippon Electric Corp. (NEC). Der **V33** bzw. μPD 70136 ist ein Nachfolger des V30. Seine Befehlsausführungszeit von 150 ns ist 3,7 mal kürzer als die des V33 und 1,6 mal kürzer als die des 80286.
Die Harris Corp. stellte kürzlich den **80C286**, wie der V33 ein Prozessor für die Klasse der AT-kompatiblen PCs, vor, der mit 20 MHz getaktet werden kann. Die bisher bekannten 80286-Prozessoren werden mit maximal 16 MHz betrieben. Der CMOS-Prozessor 80C286 ist wegen seiner geringen Leistungsaufnahme besonders gut für den Einsatz in Laptops (Schoß-PCs) geeignet.
Für die Entwicklung kostengünstiger 32-Bit-PCs gibt es jetzt den **80386SX** von Intel (S. MP 9/88). Dieser 32-Bit-Prozessor, der mit 16 MHz getaktet werden kann, ist mit einem 16-Bit-Datenbus versehen. Er läßt damit die Verwendung billigerer Peripherieschaltkreise zu, hat aber die Verarbeitungsbreite und die Softwaremöglichkeiten des 80386. Seine Geschwindigkeit liegt 10 % unter der des 80386 (16-MHz-Version). Drei renommierte PC-Hersteller kündigten bereits die Produktion von 386SX-PCs an (Compaq: Deskpro 386s, NEC und NCR Corp).
Gleichzeitig mit dem 80386SX wird auch der dazugehörige mathematische Koprozessor **80387SX** angeboten, der selbstverständlich mit den Typen 8087, 80287 und 80387 kompatibel ist. Neu von Intel ist auch die 25-MHz-Version des 32-Bit-Prozessors 80386, der **80386/25**. Auch hierfür haben bereits einige Firmen ihre ersten 25-MHz-PCs bzw. -Workstations angekündigt (Sun: Sun386i, Intel: System 302, Compaq: Deskpro 386/25, IBM: PS/2 Modell 70, ALR: Flex-Cache 25386DT, Kobil: Kobil-AT 386-25, Siemens: PCD-3TS).
Die Weitek Corp. beginnt mit der Produktion des Arithmetikprozessors **Abacus** für 80386-Rechner. Er kann in einer 20- und einer 25-MHz-Version geliefert werden. Alte Weitek-Koprozessoren benötigten bisher einen PC-Erweiterungssteckplatz, Abacus kann in einen 121poligen 80387-Sockel gesteckt werden. Nachteil: Um die Leistungsfähigkeit von Abacus voll ausschöpfen zu können, wird Weitek-kompatible Software benötigt.
MP-HK

MS-DOS und mehrere Nutzer

Quick-Connect heißt eine Multiusersoftware von Virtual Systems, mit der mehrere Nutzer an einen PC/XT, AT, 386-AT oder ein PC/2 Modell 80 angeschlossen werden können. Das Quick Connect ist unter MS-/PC-DOS ab Version 3.0 lauffähig. Die Programme Lotus 1–2–3, WordPerfect und dBase III Plus werden ebenfalls unterstützt. Jedem Nutzer stehen dabei maximal 640 KByte RAM zur Verfügung.
Eine andere Lösung fand Sun Microsystems mit den Workstations Sun386i. In diese Workstations wurde die Multiuser-Mulitasking-Window-Umgebung SunOS 4.0 implementiert, in der neben UNIX-Programmen auch mehrere DOS-Anwendungen parallel ablaufen können. Diese MS-DOS-Anwendungen, die nicht speziell für Multitasking-Betrieb ausgelegt sind, können trotzdem gleichzeitig ablaufen und das sogar ohne die geringste Änderung. Außerdem sind die Workstations voll kompatibel zu MS-DOS. Auch der Informationsaustausch (z. B. Text) zwischen UNIX und MS-DOS wird gewährleistet.
aus miniMicro-magazin 4, 6/1988 *MP*

4-MBit-RAMs vor Serienfertigung

Die Nippon Electric Corporation will Ende dieses Jahres als erste den 4-MBit-DRAM in nennenswerten Stückzahlen anbieten.
Siemens, neben Toshiba, Hitashi und Mitsubishi derzeit einer der bedeutendsten Produzenten von 1-MBit-Chips, hat im Juni damit begonnen, Muster des 4-MBit-DRAMs zur Erprobung an Kunden auszuliefern. Damit liegt das Unternehmen im Zeitplan seines 1984 gestarteten Magaprojekts. Die Serienfertigung soll im Laufe des Jahres 1989 aufgenommen werden. Die Speicherchips verfügen auf einer Fläche von nur 91 mm^2 über vier Millionen Speicherzellen. Dies entspricht einer Informationsmenge von rund 250 Schreibmaschinenseiten. Im Vergleich zum 1-MBit-DRAM bedeutet dies eine wei-

tere Vervierfachung der Speicherkapazität, obwohl die Fläche des 4-MBit-Chips gegenüber der des 1-MBit-Speichers nicht einmal verdoppelt wurde.
Schätzungen zufolge werden 1-MBit-DRAMs bis 1989 weltweit mehr als 50 Prozent aller Speichermedien ausmachen.

MP

Version 4.0 von PC-DOS und MS-DOS

Seit Ende Juli ist das PC-Betriebssystem DOS 4.0 von IBM verfügbar. Microsoft kündigte unterdessen ebenfalls die Version 4.0 von MS-DOS an. Beide Betriebssysteme unterscheiden sich von ihren Vorgängern PC-/MS-DOS 3.3 durch eine menügesteuerte Benutzeroberfläche, wie Pull-down-Menüs und kontextabhängige Hilfstexte, und sollen daher leichter zu bedienen sein. Auch der adressierbare Speicher (bisher 640 KByte), die Dateigröße (jetzt 2 GByte) und die Größe der Festplattenpartitionen (bisher mußte eine Festplatte mit mehr als 32 MByte in logische Laufwerke mit maximal je 32 MByte unterteilt werden) haben sich erhöht.
PC-DOS 4.0 läuft auf allen IBM-PCs und PS/2-Modellen und wird in 14 Sprachen angeboten. MS-DOS 4.0 ist für die IBM-kompatiblen PCs geeignet, wird von Microsoft zur Zeit jedoch nur an OEM-Partner verkauft.
aus PC-Woche 30 und 31/1988 *MP*

Abkommen über Herstellung und Vertrieb des ersten „Universal-PS/2-Chip-Set"

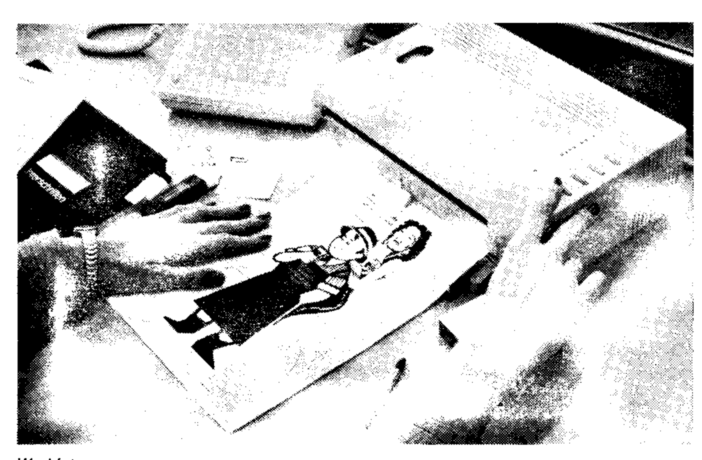
Die Bull-Gruppe mit Konzernsitz in Paris und die G-2 Inc., Tochtergesellschaft der kalifornischen LSI Logic, haben ein Technologie-Austauschabkommen im Hinblick auf die kommende Mikrocomputer-Generation getroffen. Damit sollen kostengünstige Systemangebote mit den Vorteilen und Möglichkeiten der Micro-Channel-Architektur und des Betriebssystems OS/2 möglich werden.
Das Abkommen sieht vor, daß G-2 einen von Bull entwickelten *Universal-Chip-Set* weltweit herstellen und vermarkten darf und darüber hinaus berechtigt ist, die von Bull entwickelte BIOS-Software (basic input/output system) zu vertreiben. Wie Bill 0 Meara, Präsident der G-2 Inc., anläßlich der Vertragsunterzeichnung betonte, haben der Chipsatz und BIOS als erste Produkte ihrer Art den Vorteil, „uneingeschränkt kompatibel zu den Mikroprozessoren 80286, 80386 und zum neuen 80386SX (p9)" zu sein. Als Gegenleistung erhält Bull von G-2 neben Lizenzgebühren die Technologie für das Chipdesign und eine zuverlässige Versorgung mit Chip-Sets.
Beide Unternehmen äußerten die Zuversicht, daß sich OS/2 und die Micro-Channel-Architektur rasch zum neuen De-facto-Industriestandard entwickeln würden. Ungeachtet der Unterstützung für diese neuen Konzepte bleibe aber der Support für die gegenwärtige Systemgeneration auf der Basis von DOS und PROLOGUE sowie der AT-/XT-Architektur ungeschmälert bestehen.
G-2 kündigte an, erste Muster des aus sieben Bausteinen bestehenden Universal-Chip-Set würden noch im dritten Quartal dieses Jahres verfügbar sein, und bereits im vierten Quartal würde mit der Serienproduktion begonnen werden. Bull wird zu den ersten Abnehmern von Produktionsstückzahlen gehören. *MP*

Personal Publishing auf allen Modellen des IBM PS/2

Mit der Ankündigung der PostScript-Adapterkarte für die Modelle 50, 60 und 80 des IBM PS/2 und einem neuen PostScript-Adapterprogramm ist das IBM Personal Publishing System jetzt auf allen Modellen des IBM PS/2 einsetzbar.
Das Personal Publishing System der IBM ist eine integrierte Lösung für Desktop Publishing (DTP). Es basiert auf dem Personal System/2 und einem Tischdrucker mit Lasertechnik. Kernstück des Systems ist das Layoutprogramm Pagemaker des amerikanischen Software-Hauses Aldus und das Seitenbeschreibungsprogramm PostScript von Adobe. Mittels PostScript können mit allen Programmen Texte und Bilder in einem einheitlichen Datenformat an den Drucker gegeben werden, ohne daß dieser eigens angepaßt werden muß.
Von anderen DTP-Systemen unterscheidet sich das IBM Personal Publishing System in der Implementierung von PostScript. Die Drucksteuerung erfolgt im Personal Computer selbst, was eine schnelle Übertragung von Dokumenten mit hohem Speicherplatzbedarf ermöglicht.
aus IBM-Nachrichten 6/1988 *MP*

Mini-Fax

Nur die Stellfläche von der Größe einer A4-Seite braucht der derzeit kleinste Fernkopierer von Siemens, der HF 2301. Damit paßt dieser Mini-Fax auf jeden Schreibtisch. Er arbeitet besonders geräuscharm, und durch seine ergonomische Gestaltung ist er auch einfach zu bedienen.
Der HF 2301 überträgt und empfängt über das Telefonnetz Unterlagen bis zum A4-Format detailgetreu in 16 Graustufen. Er kann im Stapelbetrieb bis zu fünf Vorlagen nacheinander ohne zusätzliche Bedienungshandgriffe versenden. Der Empfang läuft ebenfalls automatisch ab. Auch für lokales Kopieren „am Platz" läßt sich der HF 2301 verwenden – alles griffbereit auf dem Schreibtisch. *MP*



Werkfoto

Neues Material für optische Datenspeicherung

Unter der Bezeichnung „Digital Paper" wurde von der Firma ICI ein flexibles Speichermedium angekündigt, mit dem optische Speicher beliebiger Form (Streifen, Disketten, Etiketten) hergestellt werden können. Das Grundmaterial ist ein Polyestersubstrat, beschichtet mit Farbstoffen und Polymeren. Die Entwickler des Speichermediums rechnen damit, daß eine neue Generation leistungsfähiger Laufwerke entwickelt wird, die das Medium wie Papier benutzen. Die optische Speicherung auf flexiblen Medien gestattet eine wesentliche Steigerung der Speicherdichte. Auf eine Spule von 880 Meter Länge und 1,3 cm Breite sollen 600 000 Megabyte gespeichert werden können. Die Speicherdichte läßt sich durch Verbesserung der Lasertechnik weiter erhöhen. Gegenüber der Speicherung auf magnetischer Festplatte lassen sich die Kosten um den Faktor 200 verringern.
Quelle: Die Welt vom 17. 5. 88 *Wi*

Koordinierung der Supraleiterforschung in Japan

Nach Angaben des Battelle Memorial Institute sind für das japanische Programm zur Entwicklung und Implementierung der Supraleitertechnologie drei Organisationen zuständig, das Ministerium für Internationalen Handel und Industrie (MITI), die Wissenschafts- und Technologie-Agentur und das Ministerium für Erziehung.
Das Ministerium für Internationalen Handel und Industrie ist bemüht, ein Zentrum für Supraleitfähigkeit einzurichten, an dem zu 80% Unternehmen der Energieerzeugung, des Elektromaschinenbaus, der Kabel- und Keramikindustrie beteiligt sein sollen. Zu dem Zentrum soll ein Forschungslaboratorium gehören, das japanischen, amerikanischen und europäischen Wissenschaftlern im Austausch zur Verfügung stehen soll. In Zusammenarbeit mit den beteiligten Unternehmen will MITI die Forschungs- und Entwicklungsprojekte auf den drei folgenden Gebieten organisieren:
- Theorie der Supraleitung und supraleitenden Werkstoffe
- die Dünnschichtverfahren zur Herstellung von Josephson-Junction-Schaltungen und deren Anwendung auf Computer und andere elektronische Ausrüstungen
- die Wickeltechnik mit Supraleitern, Energieerzeugung mit Supraleitern und Bau von Versuchsmodellen zur Energiespeicherung.

Die Wissenschafts- und Technologie-Agentur ist für ein Projekt verantwortlich, an dem 130 Unternehmen teilnehmen und das sich mit der Theorie, Synthese und Materialbewertung der Supraleitertechnolgie befaßt. Letzteres steht besonders im Vordergrund.
Das Ministerium für Erziehung kontrolliert die Forschungsbudgets der japanischen Universitäten.
Auch außerhalb dieser Forschungsförderung investieren viele japanische Großunternehmen erhebliche Mittel zur Entwicklung von Hochtemperatursupraleitern und ihren Anwendungen. Allein bei der Sumitomo Electric Industries Ltd. arbeiten 65 Forscher an Supraleitern in Form von Drähten oder dünnen Schichten. Von 200 Erfindern wurden 700 Patente angemeldet, die sich alle auf Supraleitung beziehen.
Quellen: Außenwirtschaft. Berlin 16 (1988) 3, S. 10, 11 Blick durch die Wirtschaft vom 22. 12. 1987

Wi

Polymere mit ferromagnetischen Eigenschaften

Polymerwissenschaftler in den USA arbeiten gegenwärtig daran, neuartige Polymermischungen zu entwikkeln, mit denen es möglich wird, die bisher üblichen Ferromagnetwerkstoffe durch Polymere zu ersetzen.
Diese Polymere sollen bereits etwa 25% stärkere Magnetkräfte aufweisen als herkömmliche Magnete auf Eisenbasis. Diese Werte wurden aber nur im Tiefsttemperaturbereich um 5 Kelvin erreicht.
Jetzt wird daran gearbeitet, mit neuartigen Polymerverbindungen und speziellen Herstellungstechniken, die magnetischen Eigenschaften auch bei Raumtemperatur oder bei Temperaturen um 0 °C zu erreichen. Bei den neuen Verbindungen handelt es sich um polymere Ferrocene (organische Übergangsmetall-Komplexe) in Verbindung mit Tetracyanoethylenen. Das sind komplex gebaute, organische Verbindungen, bei denen das Eisenatom nicht mit Kohlenstoffatomen des organischen Anteils aus Cyclopentadienyl verbunden ist.
Sollte es gelingen, derartige Kunststoffe auch bei Temperaturen um mehr als 100 °C magnetisch zu halten, eröffnen sich zahlreiche Anwendungen zur Herstellung preiswerter Magnete wie Folien oder Filme für magneto-optische Datenspeicher sowie für Magnetköpfe.
Quelle: Die Welt vom 19. 3. 1988 *Wi*

NEC Corp. stellt neuen Supercomputer vor

Die neue Supercomputerserie SX A der Firma NEC Corp. soll ein äußerst günstiges Kosten-Leistungs-Verhältnis aufweisen und vier Modelle umfassen, deren Rechengeschwindigkeit wie folgt gestuft ist: 1,3 Mrd., 665 Mio, 330 Mio und 250 Mio Gleitkommaoperationen/Sekunde.
Die Verarbeitungskapazität des Steuergerätes wurde im Vergleich zu bisherigen Geräten um den Faktor 1,9 erhöht. *ADN-Wi*

Einführung in die Informationsverarbeitung

von G. und L. Entress, VEB Verlag Technik, Berlin 1986, 311 Seiten, 23,50 M

Das Grundlagenwissen auf dem Gebiet der Informationsverarbeitung wird, bedingt durch die stürmische Entwicklung der Computertechnik, zunehmend für breite Kreise von Technikern und Ingenieuren unentbehrlich. Es ist daher begrüßenwert, daß in diesem Buch eine umfassende Sammlung von technischen Wirkprinzipien der Informationsverarbeitung in verständlicher Form zusammengestellt wurde.

Die Autoren erläutern nach einer kurzen, motivierenden Einführung und Darstellung der allgemeinen Grundlagen den Digitalrechner mit seinen internen Bestanteilen als wichtigstes Mittel der Informationsverarbeitung. Kapitelweise stellen sie dabei jeweils Grundaufbau und Organisationsprinzip von Arbeitsspeicher, Prozessor, Zentraleinheit und Peripherieanschluß anhand instruktiver verallgemeinerter Bilder und Tafeln dar. Jedes Kapitel schließt ab mit einer Reihe von Kontrollfragen, die den Leser befähigen, seinen erworbenen Kenntnisstand selbständig zu überprüfen und anhand der ausführlich kommentierten Lösungen im Anhang zu beurteilen. Dem gleichen Aufbau folgen auch zwei weitere Kapitel, die sich der Umgebung des Rechners, d. h. den peripheren Geräten und den Datenübertragungsmechanismen, zuwenden.

Vorwiegend Übersichtscharakter haben die anschließenden Kapitel zur Bauelementebasis, zu den Informationsverarbeitungssystemen – mit der Gliederung: Einrechnerkonzepte, Mehrrechnerkonzepte, Rechnerklassen, Diagnosesystem, Rechnergenerationen und Programmierung – und zu Einsatzbeispielen.

Hier vermutet man zunächst aufgrund der zahlreichen Gliederungspunkte (und ohne Beachtung der Tatsache, daß deren Anzahl zur Seitenzahl dieser Kapitel im Verhältnis 1,5 : 1 steht) mehr Informationen zu den angesprochenen Schwerpunkten, muß jedoch feststellen, daß diese nur kurz ausgeführt werden. Schwerpunktbezogene Hinweise auf weiterführende Literatur wären in dieser Situation wünschenswert, würden das Buch um eine weitere positive Gebrauchseigenschaft bereichern und der detaillierten Gliederung ihre Daseinsberechtigung geben.

Grundlegende Betrachtungen zur Betriebssicherheit von Systemen und zur Einordnung der Teildisziplinen der automatischen Informationsverarbeitung runden den Übersichtsteil ab. Wertvoll für den Leser sind die Zusammenstellungen von gebräuchlichen Kodierungsvorschriften und Abkürzungen sowie ein ausführliches Sachwortverzeichnis.

Alles in allem gibt das Buch einen leicht verständlichen Einstieg in die Informationsverarbeitung und vermittelt ein entsprechendes Überblickswissen zu diesem Fachgebiet.

Dr. H.-D. Wuttke

In FORTH denken

(Thinking FORTH)
Von L. Brodie, Carl Hanser Verlag München, Wien, Prentice Hall International Inc. London, 1986, 290 S.

Programmieren als kreativer Akt vollzieht sich in umfangreichen und komplizierten Denkprozessen, derer man sich häufig nicht bis ins letzte bewußt ist. Diese Prozesse stehen im Mittelpunkt des zweiten Buchs von Leo Brodie. Nach „Programmieren in FORTH" liegt nun eine stärker methodisch orientierte Darstellung praktischer Fragen der Softwaretechnologie, gespiegelt an den Konzepten von FORTH, vor.

Im Vordergrund steht die Frage, wie Denkprozesse beim Entwerfen, Implementieren und Testen von Programmen durch geeignete Strategien unterstützt und auf inhaltliche Aspekte zu bearbeitender Aufgabenstellungen orientiert werden können. Brodie zeigt, daß die durch FORTH implizierte hochgradig modulare Gestaltung von Software und die damit verbundenen Entwurfsprinzipien auch außerhalb von FORTH anwendbar sind. Im einzelnen behandelt er u. a.:

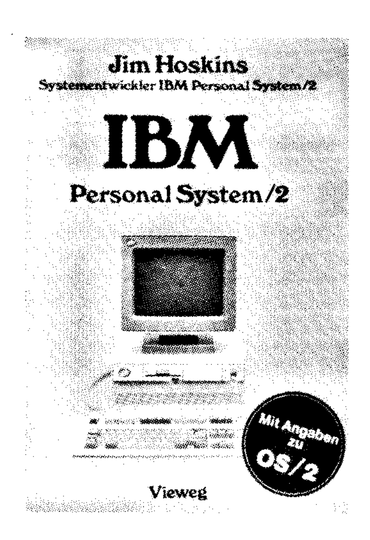
– Die FORTH-Philosophie – modulare Programmierung, Verdecken von Implementierungsdetails, Hoch-, Beschreibungs- und maschinennahe Sprachen
– Analyse – 9 Phasen des Programmierkreislaufs, Inhalt und Erarbeitung einer Spezifikation
– Grobentwurf/Dekomposition – Dekomposition nach Komponenten, Schnittstellenspezifikation, Dekomposition und Komplexität, Ebenen
– Feinentwurf/Problemlösung – Problemlösungstechniken, Syntax von FORTH, Algorithmen und Datenstrukturen
– Implementation – FORTH-Stilelemente: Gestaltung von Quelltexten, Kommentierungen, Namenskonventionen
– Faktoring – Techniken und Kriterien der Zerlegung und Zergliederung komplexer Funktionen
– Umgang mit Daten – Stacks und Zustandsinformationen
– Minimierung von Steuerstrukturen.

Die Darstellung erfolgt angenehm locker, ergänzt durch eine Vielzahl von Empfehlungen für die Programmierpraxis sowie durch relevante Programmbeispiele. Für den FORTH-Programmierer von besonderem Wert ist die Darstellung des DOER/MAKE-Konzepts (Ausführungsvektoren), mit dem die Implementierung von Programmen möglich ist, die zur Laufzeit ihr Verhalten bzw. sogar ihre Struktur ändern.

Die Monographie wird ergänzt durch eine Reihe von Interviews, in denen (FORTH-) Programmierer aus eigenem Erleben ihre Erfahrungen und Vorgehensweisen bei der Gestaltung von Software erläutern. FORTH-Nutzer mögen den Titel insgesamt mit weitestgehend ungeteilter Zustimmung aufgenommen haben:

Er gibt Antworten auf methodische Fragen, die auch aus Sicht der Informatik an FORTH zu stellen sind. Aber selbst Softwarespezialisten, die andere Programmiersprachen nutzen, können durch dieses Buch angeregt werden, ihre eigene Programmierpraxis zu überdenken.

G.-U. Vack



IBM Personal System/2

Beschreibung – Einsatz – Anwendung – Technische Details. Von Jim Hoskins, Vieweg 1987, 300 S., 64,– DM, ISBN 3-528-04419-5

Mich reizte es sofort, dieses Buch zu lesen, weil es erstens eine zusammenfassende Darstellung des gesamten Personal System/2 (PS/2) von IBM zu versprechen schien und weil es zweitens von einem Mitglied des Entwicklungsteams geschrieben wurde. Erstaunt mußte ich dann gleich in der Einleitung feststellen, daß ausgerechnet ein Entwickler wie Jim Hoskins nicht ein Buch für Insider der PC-Branche, sondern für alle potentiellen Käufer geschrieben hat. Diese Käufer verfügen natürlich in der Mehrheit nur über ein Computer-Querschnittswissen, wenn nicht sogar über gar keine Vorkenntnisse. Aber gerade diese drei Gruppen von PC-Nutzern, die Computerspezialisten, die Computerbediener und die Einsteiger, versucht Hoskins unter einen Hut zu bringen.

Er fängt erst einmal ganz konventionell bei der Entwicklungsgeschichte des IBM-PC im August 1981 an und stellt fest, daß man sich damals noch mit dem Prozessor 8088, 16 KByte RAM und einer 160-KByte-Floppy begnügte. Mit Fotos vom PC/XT, dem AT und den PS/2-Modellen 30, 50, 60, 80 und Convertible kommt er zu den Konstruktions- und Leistungsunterschieden zwischen Personal Computer und Personal System.

Für die Einsteiger und Computerbediener erklärt er Begriffe von ROM = Read Only Memory bis AIX = Advanced Interactive Executive (UNIX-kompatibles Betriebssystem des PC/RT und des PS/2 Modell 80). Er vergleicht 4 GByte mit 2 Millionen Schreibmaschinenseiten, die einen 70 Stockwerke hohen Stapel ergeben. Mit den mundgerechten Erklärungen der ersten Handgriffe nach dem Einlegen der Referenzdiskette kann sich ein Neuling an das PS/2 setzen und anfangen. Das schafft Erfolgserlebnisse.

Dem potentiellen Käufer führt der Autor die gesamte Peripherie, die Einbindung der Rechner in verschiedene Netze oder ihren Einsatz als Mainfraim-Gateway (Bindeglied zwischen IBM-Großrechner und PC-Breitband-Netzwerk) vor. Letztlich bietet er sogar Workstation-Konfigurationen einschließlich Software für verschiedenste Aufgaben an.

Trotzdem bleiben auf den insgesamt 300 Seiten, von denen durchschnittlich jede zweite mit einem Bild oder einer Tabelle illustriert ist, genug Informationen für den Profi übrig. Hoskins versucht mit Grafiken den Qualitätssprung vom PC zum PS/2 aufzuzeigen. Die neue Qualität des PS/2 besteht in der hard- und softwareseitigen Erweiterbarkeit des Systems. Immerhin kann beispielsweise das Mittelklasse-Modell 60 (Tower mit 80286-Prozessor) mit RAM-Karten auf 15 MByte erweitert werden. Zusätzlich kann es mit 2 Festplattenlaufwerken 185 MByte verwalten, virtuell läßt der Prozessor sogar 1 GByte zu. Nicht weniger interessant sind die Angaben zur Grafikfähigkeit des PS/2, die (wie alle anderen Angaben auch) mit den Parametern des PC/XT und AT verglichen werden. Das standardmäßig auf der Systemplatine mitgelieferte VGA (Video Graphics Array) erzeugt 640 × 480 Bildpunkte mit 16 Farben. Mit dem Bildschirmadapter 8514 A sind sogar 1024 × 768 Bildpunkte mit 256 Farben möglich.

Ausführlich geht Hoskins auch auf die Software-Kompatibilität zu den PCs ein. Abgesehen von der Verwendung der 3,5-Zoll-Floppylaufwerke ist das Operating System/2 (OS/2) der Modelle 50 bis 80 weitgehend kompatibel zum PC-DOS. Im Real Mode arbeiten die PS/2-Modelle unter PC-DOS 3.3, während die Modelle 50 bis 80 im Protected Mode unter OS/2 bis zu 12 Tasks gleichzeitig ablaufen lassen können (Eine Task kann dabei auch PC-DOS sein!).

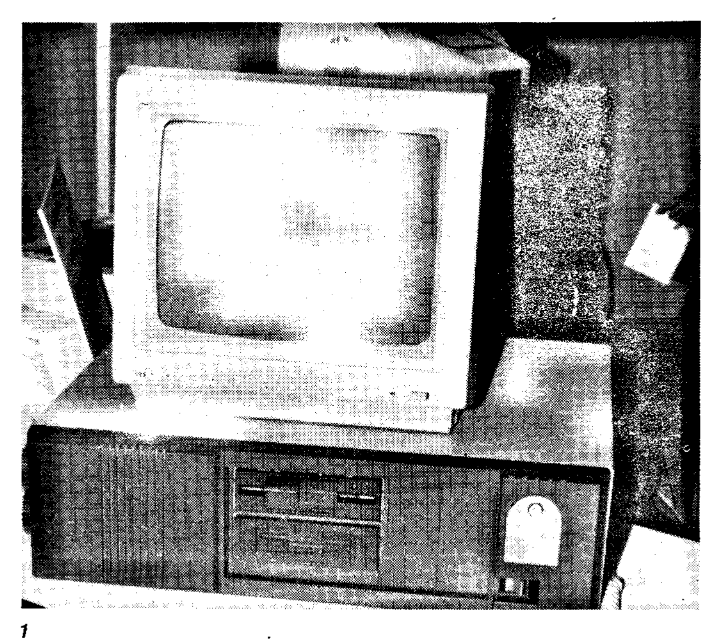
Natürlich bin ich bei diesem Buch auch an seine Grenzen gestoßen, da sich Hoskins bei seiner Zielstetzung irgendwo einschränken mußte. Er erwähnte z. B. die Modelle 30 und Convertible (tragbares PS/2) nur am Rande, die Modelle 25 und 70 sind (noch) nicht vertreten. Er beschränkt sich auf die Modelle mit der neuen Mikrokanal-Architektur. Das Wesen dieser Architektur behandelt er jedoch nur sehr oberflächlich. Bei der Vorstellung der PS/2-Peripherie macht Hoskins (außer bei Druckern und Maus) keine Aussagen zur Kompatibilität mit Geräten anderer Hersteller.

Trotzdem bin ich der Meinung, daß der Autor mit seinem Buch das eingangs erwähnte Ziel erreicht hat. Er führt den Einsteiger direkt (ohne den Umweg über kleine Computersysteme) an modernste PCs heran. Unterstützt wird er natürlich von der Tatsache, daß neue Rechner höherer Komplexität mit immer bedienfreundlicherer Software ausgerüstet werden.

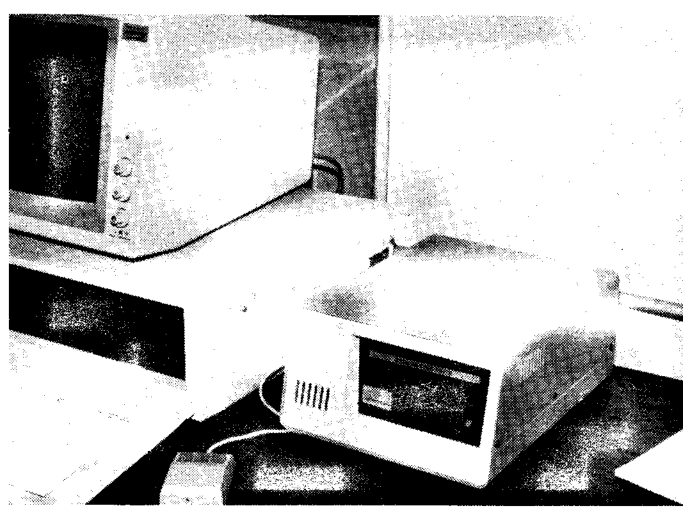
MP-Hk

87. Budapester Internationale Messe

Vom 18. bis 26. Mai 1988 fand in Budapest die 87. Internationale Messe statt, an der sich über 2000 Aussteller aus 31 Ländern beteiligten.
Zu den vorrangigen Zielen dieser Technischen Messe zählten aus ungarischer Sicht nicht nur die Vorstellung von Erzeugnissen, die als Motor der technischen Erneuerung gelten, sondern auch die Darstellung der bisherigen und zu erwartenden Ergebnisse der marktorientierten Entwicklungsprojekte und die Präsentation einer Reihe exportfähiger Produkte ungarischer Firmen. Gleichzeitig sollten die umfangreichen Importmöglichkeiten aufgezeigt werden. Unter Berücksichtigung der wirtschaftlichen Zielsetzungen Ungarns lag der Schwerpunkt deshalb bei den Bereichen, die für die Umstrukturierung der Wirtschaft besonders wichtig sind, vor allem bei der Elektronik und elektronischen Industrie. Computer- und Nachrichtentechnik sowie Energetik gehörten somit zu den bedeutendsten Warengruppen der Messe, was sich auch in der Vielzahl der Exponate ausdrückte. Der im folgenden versuchte Überblick zur Computertechnik muß leider lückenhaft bleiben; zu viele Betriebe aller Eigentumsformen boten Hard- und Software an als daß alle berücksichtigt werden können.
Als wesentlicher Eindruck bleibt festzustellen, daß im Bereich Computertechnik viele Betriebe – darunter sehr viele kleine – sehr leistungsfähige und fortschrittliche Technik anbieten, dabei allerdings oft lediglich als Zwischenhändler fungieren.
Eine seit 1981 expandierende Genossenschaft ist Müszertechnika Kisszövetkezet (MT), Produzent auch von IBM-PC-kompatiblen 32-Bit-PCs. Zur diesjährigen Messe war MT mit einem zum PS/2 Modell 80 teilweise kompatiblen PC vertreten (ohne den für das IBM-Modell 80 charakteristischen Mikrokanal). Der **MPS 386** – Fotografieren nicht erlaubt! – hat die gleichen äußeren Abmessungen wie das Original, verfügt über den 80386-Prozessor mit 16 MHz, 1 MByte RAM, 3,5-Zoll-Floppy-Laufwerke mit 1,4 MByte, 96-MByte-Festplatte, VGA-kompatible Grafik, RS-232- und Centronics-Interface und läuft derzeit unter DOS 3.3.
Auch die meisten anderen ungarischen Anbieter hatten neben XT- und AT-kompatiblen Modellen bereits 32-Bit-PCs ausgestellt, so daß man – oberflächlich betrachtet – hätte annehmen können, 32-Bit-Prozessoren in PCs seien inzwischen gang und gäbe. Hier sollte man jedoch nicht die tatsächliche Verfügbarkeit in der Wirtschaft außer acht lassen. Immerhin zeichnet sich damit ein Trend ab, den wir im folgenden besonders berücksichtigen wollen.
Das Forschungs- und Entwicklungszentrum für Rechentechnik SzKI – Hersteller der bekannten PROPER-Computer – stellte den neuen 32-Bit-PC **PROPER-132** gleich in verschiedenen Anwendungslösungen vor. Eine war ein Desktop-Publishing-System (DTP) auf Basis des Layout-Programms Ventura Publisher von Rank Xerox, das an die Besonderheiten der ungarischen Sprache angepaßt wurde (ebenfalls auf dem Ventura Publisher basierte übrigens ein von der oben bereits erwähnten Firma MT angebotenes DTP-System). Zur Konfiguration gehörten der Laserdrucker Rank Xerox 4045, ein Scanner von Microtek sowie ein Monitor, der im Hercules- oder EGA-Modus betrieben werden kann (Farbbild 1; alle Farbbilder siehe 3. Umschlagseite). Eine weitere Lösung zeigte den PROPER-132 in dem lokalen Netz PRONET, wofür er beispielsweise mit einer X.25-Adapterkarte ausgestattet wird (Farbbild 2). Der PROPER-132 ist ein IBM-PC/AT-kompatibler PC, der auf dem Prozessor 80386 basiert (16 MHz) und mit 1 MByte RAM, 1,2 MByte Floppy-Kapazität und 40 MByte Festplattenkapazität ausgerüstet ist.
Seit etwa einem Jahr produziert die Firma dataplan IBM-kompatible PCs; sie bot als Neuheit ebenfalls einen 32-Bit-PC an (Bild 1). Der **dp-386** besitzt einen 80386-Prozessor, der wahlweise mit bis zu 20 MHz betrieben werden kann. Zu einer hohen Verarbeitungsgeschwindigkeit trägt auch die 80-MByte-Festplatte mit einer Zugriffszeit von nur 28 ms bei. Die Floppy-Kapazität beträgt bei 5 1/4 Zoll 1,2 MByte, bei 3 1/2 Zoll 0,7 MByte. Als Option ist ein Streamer mit 60 MByte lieferbar. Besonders hervorgehoben wurde beim dp-386 die Eignung als Fileserver in Novell-Netzen, die in Ungarn offenbar bevorzugt verwendet werden.
Weitere Firmen, die 32-Bit-PCs offerierten, waren auch Controll mit dem **MC 386** (80386 mit 16 MHz, 2 MByte RAM, 1,2 MByte Floppy, 40 bis 80 MByte Festplatte), Azsio mit dem **386/W80/M** (80386 mit 20 MHz, 2 MByte RAM, 1,2 MByte Floppy, 80 MByte Festplatte) und Microsystem mit dem **PC 820-AT/1** (80386 mit 16 MHz, 1 MByte RAM, 1,2 MByte Floppy, 50 MByte Festplatte).
Die Genossenschaft Cobra, ebenso wie die meisten vorgenannten Betriebe Anbieter von Hardware und Anwendungssoftware, zeigte neben einem 32-Bit-PC den 10-kg-Portable Walkom **LCD-286** (Farbbild 3) mit 80286-Prozessor und 1 MByte RAM. Weitere Daten: Zwei 5 1/4-Zoll-Floppy-Laufwerke mit 1,2 MByte oder 3 1/2-Zoll-Laufwerke mit 1,44 MByte oder je 1 Floppy- und 1 Festplattenlaufwerk; LCD-Bildschirm mit 640 × 200 Punkten und Hintergrundbeleuchtung; 6 Erweiterungssteckplätze, davon 3 Slots für den Anwender. Dieser Portable ist als LCD-386 auch lieferbar mit 80386-Prozessor (16 MHz) und 2 MByte RAM. Die anderen Daten entsprechen im wesentlichen dem Modell LCD-286.
An dieser Stelle muß wohl vermerkt werden, daß insbesondere die kleineren Betriebe von der Situation auf dem internationalen Bauelemente-



1



2



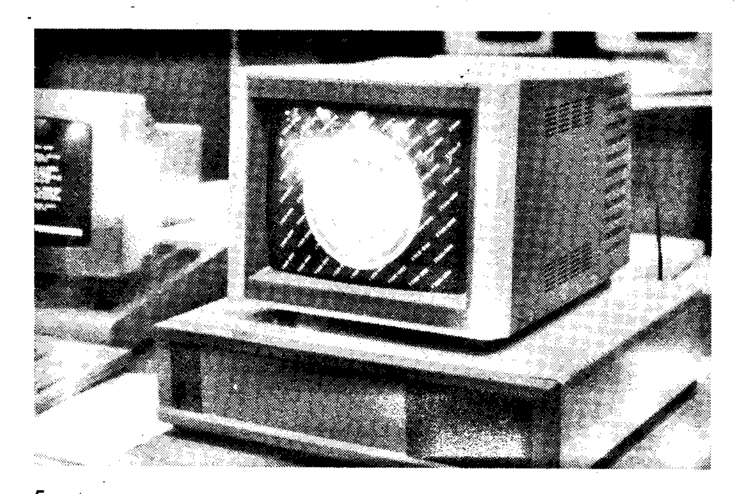
3

markt stark abhängig sind, so daß der gewählte Begriff „lieferbar" unter Umständen nur bedingt zutrifft.
Der größte ungarische Computerhersteller – Videoton – war ebenfalls erstmals mit einem 32-Bit-PC vertreten, dem **VT 180**. Während der Computer zur Messe im Tower-Gehäuse untergebracht war, zeigt ihn ein Prospekt-Foto auch als Auftischgerät in der bekannten kompakten PC-Form. Die ausführlichen Daten können unserer Beschreibung des VT 180 auf der 2. Umschlagseite entnommen werden.

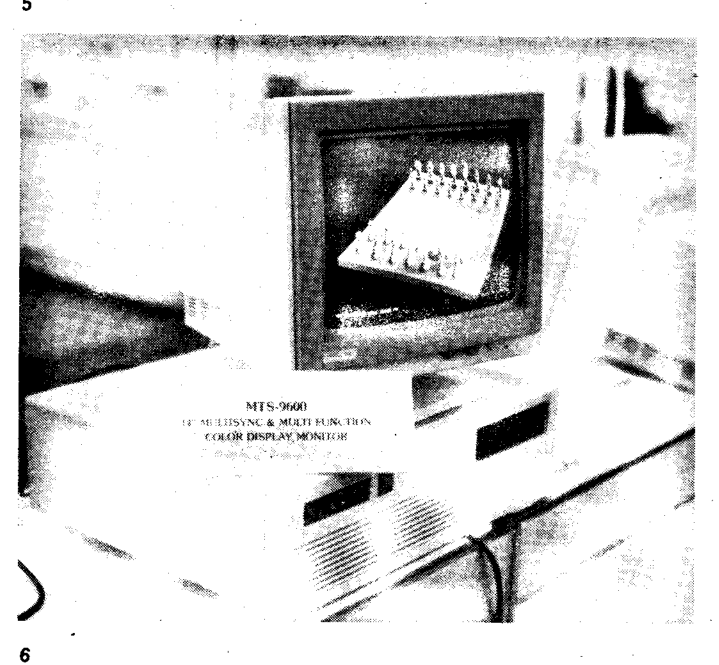
Weitere erwähnenswerte Exponate bei Videoton waren die Weiterentwicklung des VT 32 – der **VT 320** (Farbbild 4) mit 68010-Prozessor – und der optische Plattenspeicher **525 WC** (Bild 2). Das Gerät nutzt WORM-Platten (WORM – write once, read many), die mit Anwenderdaten einmal beschreibbar sind. Nach den Angaben des Standpersonals soll die genutzte 5 1/4-Zoll-Platte eine Kapazität von 120 MByte haben. Angeschlossen werden kann das Optical Disk Drive an Personal Computer ab XT-Niveau mit 20 MByte Festplatte



4



5



6

und 1 MByte Floppy. Als Betriebssystem wird in dieser Konfiguration IS-DOS verwendet.
KFKI, das Zentrale Forschungsinstitut für Physik der Ungarischen Akademie der Wissenschaften, produziert bereits seit mehreren Jahren mit der Computerfamilie TPA 11/500 32-Bit-Supermikrocomputer in kleineren Stückzahlen vor allem für den CAD/CAM-Einsatz. Ein neues Mitglied in dieser Reihe ist das als TPAstation bezeichnete Modell **TPA 11/510** (Bild 3) mit verringerter Leistung als preisgünstiger Konstrukteursarbeitsplatz. Die auf VLSI-Schaltkreisen basierende 32-Bit-CPU erbringt dennoch eine Leistung von 0,9 MIPS. Die vielfältigen Konfigurierungsmöglichkeiten erlauben den Ausbau des RAM auf 16 MByte, der Floppy-Laufwerke bis 1200 KByte, der Festplattenlaufwerke bis 1000 MByte und der Streamer bis 70 MByte. Der Grafik-Monitor hat eine Auflösung von 1024 × 864 Punkten. Als Betriebssystem wird das MOS-VP verwendet.
Von der TRITON-Cooperative wurde mit dem Telexcomputer **gepard 16** (Farbbild 5) einer interessanten Tendenz auf dem internationalen Markt Rechnung getragen: dem Aufrüsten eines PCs mit einer Leiterkarte, um Funktionen eines Telex-Gerätes simultan zur Arbeit als Personalcomputer ausführen zu können. Als Telex-PC eignet sich jeder MS-DOS-kompatible PC mit mindestens 512 KByte RAM, Bildschirm, Floppy und Drukker. So war auch der robotron A 7150 auf dem Stand als Telex-PC zu sehen.
Die Vorstellung von Erzeugnissen ungarischer Aussteller soll abgeschlossen werden mit einem neuen Produkt der MOM-Werke, dem Blattleser **ML 001** (Bild 4). Der Scanner ist in der Lage, mittels CCD-Zeilen A4-Vorlagen mit einer Auflösung von 300 × 300 dpi (dpi – dots per inch, Punkte je Zoll) einzulesen, wobei Kontrast und Lichtstärke in jeweils 14 Stufen regelbar sind. Als Interface zum Computer gibt es die serielle RS-232C- und eine parallele TTL-kompatible Schnittstelle.
Eine Vielzahl von Computerproduzenten sozialistischer und kapitalistischer Länder bot auch in Budapest ihre Erzeugnisse an. Das Kombinat Robotron beispielsweise konzentrierte sich auf das Angebot der PCs EC 1834 und A 7150 (Farbbild 6) – letzterer in der DTP-Anwendung mit Rank Xerox Ventura Publisher sowie als Programmierarbeitsplatz für Drahterodiertechnik – und von Schreibtechnik. Während der Messe vereinbarte Robotron in großem Umfang Lieferungen für Rechen- und Schreibtechnik für 1988 und 1989.
Auf dem großzügigen bulgarischen Ausstellungskomplex fiel vor allem das umfangreiche Angebot von Software-Anwendungslösungen auf. Interessant war auch die Präsentation eines 32-Bit-PCs, der nicht auf dem Intel 80386 basierte, sondern auf dem Motorola MC 68020, ergänzt um die MMU und den Koprozessor MC 68881. Der **CSY 68** (Bild 5) hat eine VMEbus-Architektur, womit er für technische Anwendungen gut geeignet ist. Dazu trägt auch das intelligente grafische Subsystem bei. 2 MByte RAM, 5 1/4-Zoll-Festplatte mit 130 MByte, Floppy mit 360 KByte bzw. 1 MByte und ein Streaming-Bandlaufwerk von 60 MByte sind weitere Merkmale. Das Multitasking-multiuser-Betriebssystem UNICSY ist UNIX-ähnlich.
IBM stellte neben dem Einstiegsmodell zur mittleren Datentechnik, dem System/36, Mod. 5363, auch ein Desktop-Publishing-System vor, bei **IBM Personal Publishing System** genannt. Dazu wurde das PS/2-Modell 30 mit einer zusätzlichen Leiterkarte ausgestattet, die einen Motorola-68000-Prozessor, 2,5 MByte Speicher und ein Adapterprogramm für die Seitenbeschreibungssprache PostScript enthält. Die Gestaltung der Seiten wird mit dem Layoutprogramm PageMaker bewerkstelligt. Zur Konfiguration gehörten noch ein Scanner 3117 und ein Laserdrucker 4216.
Auch Hewlett-Packard zählt zu den Anbietern für das „Publizieren am Schreibtisch". Gezeigt wurde das **DTP-System** mit dem HP-Vectra-PC, Scanner HP 9190A und LaserJet-Drucker (Farbbild 7). Als Gestaltungsprogramm werden PageMaker und MS-Windows verwendet, zusätzlich PostScript-Driver und Font von Hewlett Packard. Da die schon recht hohe Auflösung von 300 × 300 Punkten pro Zoll bei Laserdruckern für wirklich professionelle Anwendungen weitaus zu gering ist (Fotosatzanlagen liefern über 2000 × 2000 Punkte pro Zoll), können über einen Raster Image Processor (RIP2) mit Post-Script-Fonts die gestalteten Seiten auch auf einer Satzanlage Linotronic 100/300 ausgegeben werden.
Die auf der Leipziger Frühjahrsmesse beobachtete verstärkte Präsenz Taiwans wurde auf der Budapester Messe noch deutlicher. Das Angebot reichte von Zusatzplatinen über OEM-Baugruppen, Monitoren bis zu 32-Bit-PCs. Die Monterey International Corp. beispielsweise zeigte sowohl 16-Bit-PCs mit 8088- und 80286-Prozessor als auch das Spitzenmodell **MS-3220 Tower 386** mit dem 16/20-MHz-Prozessor 80386 (Farbbild 8).
Als schnellste 80386-Maschine der Welt präsentierte die Tecmate Electronic Inc. ihr **System/386** (Farbbild 9). Das mit 80386-CPU (20 MHz) und 80287-Koprozessor ausgestattete Modell soll 20 Prozent schneller als ein Compaq Deskpro 386 und fast so schnell wie ein VAX 11/780-Minicomputer sein – zu weniger als 1/10 des Preises von Minicomputern. Der RAM ist von 1 MByte bis 4 MByte erweiterbar, der Cache hat 64 KByte. Das System umfaßt 5 Modelle; während das kleinste unter DOS 3.3 arbeitet, steht für die anderen Modelle das Multiuser-multitasking-Betriebssystem PC-MOS/386 zur Verfügung, womit bis zu 25 Stationen (User) an ein Gerät angeschlossen werden können.
GEFO Inc. demonstrierte eine Vielzahl von Monitoren, unter anderem, um die Eignung für das PS/2 von IBM hervorzuheben. Ein Beispiel dafür ist der MTS-9600 (Bild 6), der sich als Multisync-Farbmonitor nicht nur selbständig auf die gängigen Adapter für CGA, EGA und PEA, sondern auch auf den PS/2-Standard VGA einstellt (VGA – Video Graphics Array). Der Monitor ist für analoge und TTL-Ansteuerung geeignet und liefert auf dem 14-Zoll-Bildschirm (240 mm × 180 mm) mit einer Horizontalfrequenz von 15–35 KHz und Vertikalfrequenz von 50–70 Hz ein flimmerfreies Bild in bis zu 256 000 Farben.

Text und Fotos: Hans Weiß




1



2

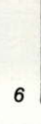





3

4

5





6



7



8



9

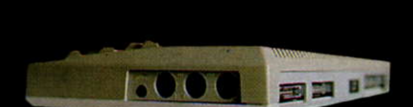
INTERNATIONALE MESSE

87. BUDAPESTER

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 10
S. 289–320



Computergrundgerät



Anschlußbuchsen und Steckverbinder am Computergrundgerät



Gesamtgerät A 5105

Diskettenspeichereinheit mit Steckverbindern zum Computergrundgerät





Rückwand der Diskettenspeichereinheit

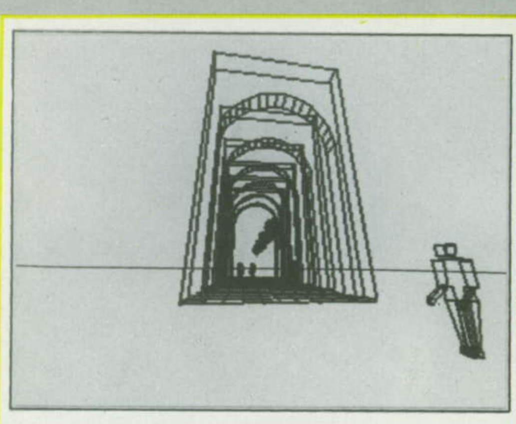
Datenvergleich Kleincomputer/Bildungscomputer

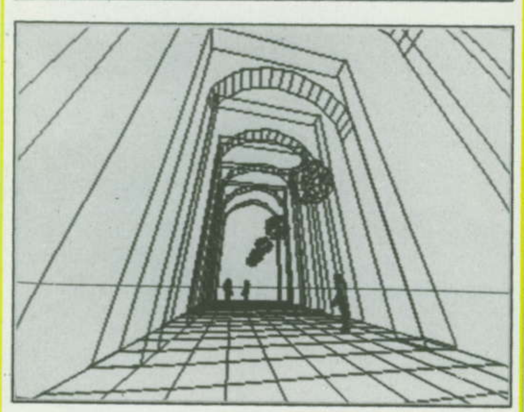
Parameter	KC 85/3	KC 87	BIC A 5105
Konfiguration	Basisgerät Tastatur Module	Kompaktgerät Module	Computergrundgerät Diskettenspeichereinheit Monitor
CPU	UB 880 D	UB 880 D	UA 880 D
Taktfrequenz	1,75 MHz	2,5 MHz	3,75 MHz
Speicherausbau – Programmspeicher – Arbeitsspeicher – Bildspeicher – Massenspeicher	 16 kB 16 kB 16 kB Magnetbandgerät	 18 kB 16 kB 2 kB Magnetbandgerät	 48 kB 64 kB 64 kB × 16 bit Magnetbandgerät Diskettenspeicher 1.6
Tastatur	schreibmaschinenähnlich	schreibmaschinenähnlich	Computerflachtastatur
Anzeigegerät	Fernsehgerät	Fernsehgerät	Monitor Fernsehgerät
Bildaufbau	40 Zeichen × 32 Zeilen Vollgrafik 320 × 256 Bildpunkte	40 Zeichen × 24 Zeilen Symbolgrafik	40 Zeichen × 25 Zeilen 80 Zeichen × 25 Zeilen Vollgrafik 640 × 200 Bildpunkte 320 × 200 Bildpunkte
Betriebssystem	CAOS V 3.1	Z 9001	RBASIC (ROM) SCPX 5105 (RAM)
Interface	Magnetbandinterface 2 Modulsteckplätze Expansionsinterface	Magnetbandinterface 4 Modulsteckplätze –	Magnetbandinterface 1 Modulsteckplatz BUS-Anschluß
Interfaces	– – HF, Audio-, Video-, RGB-Ausgang – – –	2 Steuerhebel parallel Interface (8 bit, 1 × CTC-Eingang, 1 × CTC-Ausgang) HF-, RGB-Ausgang – – –	2 Steuerhebel 2 × parallel Interface (wie KC 87) Audio-, Video-, RGB-Ausgang HF über Modulatorbaugruppe Lokalnetzanschluß Druckeranschluß (V.24 unidirektional Plotter-anschluß) (V.24 bidirektional) ext. Diskettenlaufwerk

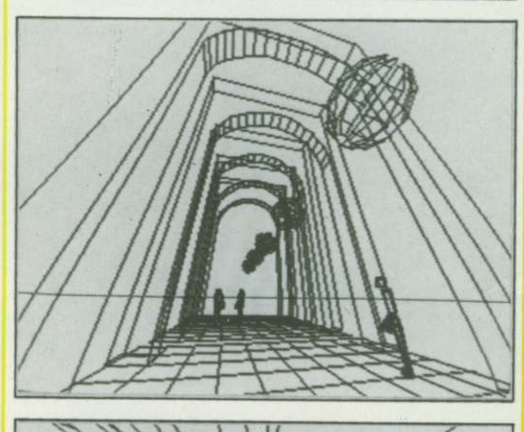

ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK
MP
Heft 11 · 1988
Mikroprozessortechnik
VEB Verlag Technik Berlin
ISSN 0232-2892
EPROMs –
größere Speicher
schneller programmierbar
Rekursion –
ein Baum
voller Menüs
ASIC
Jedem Entwickler
seinen Schaltkreis

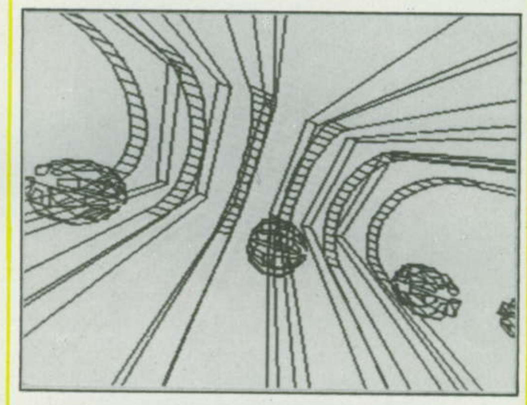
Bildserie 1 Testbeispiel „Stadt": Grundriß, Isometrie, Perspektiven mit variierter Augpunkt-, Zielpunkt-, Brennweiteneinstellung

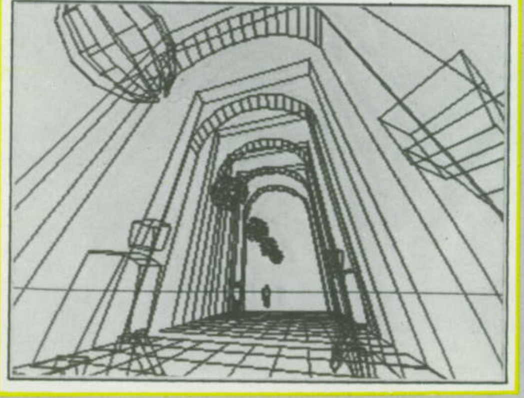
Bildserie 2 Testbeispiel „Gewölbe": Phasen einer Bewegungssequenz (fixierter Zielpunkt, manipulierter Augpunkt, Weitwinkelobjektiv)





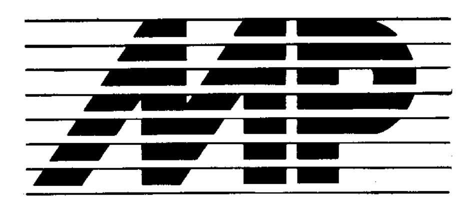






Lesen Sie dazu unseren Beitrag „3D-Simulation – interaktiver Entwurf von räumlichen Modellen".

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 11 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2870371); Herbert Hemke, Redakteur (Tel. 2870203); Sekretariat Tel. 2870381

Gestaltung Christina Bauer

Titelfotos Peter Kalbe

Beirat Dr. Ludwig Claßen, Dr. Heinz Florin, Prof. Dr. sc. Rolf Giesecke, Joachim Hahne, Prof. Dr. sc. Dieter Hammer, Prof. Dr. sc. Thomas Horn, Prof. Dr. Albert Jugel, Prof. Dr. Bernd Junghans, Dr. Dietmar Keller, Prof. Dr. sc. Gernot Meyer, Prof. Dr. sc. Bernd-Georg Münzer, Prof. Dr. sc. Peter Neubert, Prof. Dr. sc. Rudolf Arthur Pose, Prof. Dr. sc. Dr. Michael Roth (Vorsitzender), Dr. Gerhard Schulze, Prof. Dr. sc. Manfred Seifart, Dr. Dieter Simon, Dr. Rolf Wätzig, Prof. Dr. sc. Jürgen Zaremba

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

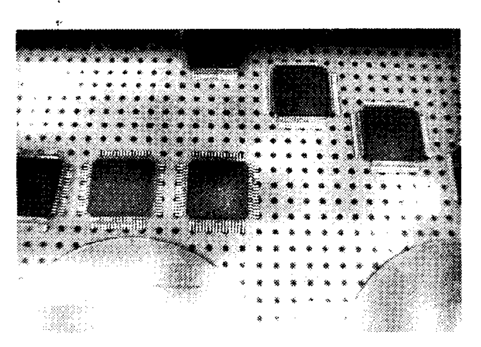
Redaktionsschluß: 13. September 1988

AN (EDV) 49837

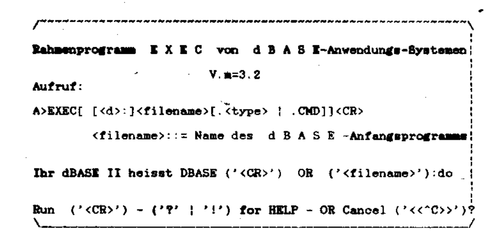
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

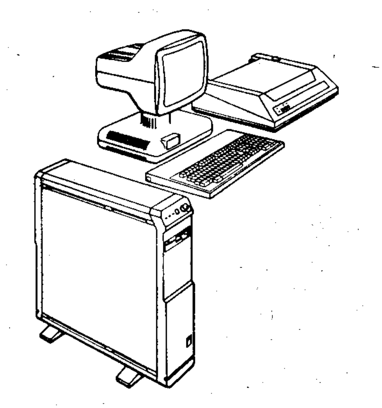
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propagandit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tisku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Előfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Há Nôi; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig



Mit unseren Beiträgen zum Thema ASIC auf der **Seite 323** beginnen wir mit einer Beitragsfolge, die die ASICs, deren Entwurf, ihren Einsatz sowie die in der DDR verfügbaren Entwicklungssysteme beinhaltet.



Auf der **Seite 331** finden Sie den Artikel „EXEC – ein Startprogramm für dBASE II". Er ist der zweite Teil unserer Beitragsfolge zum Thema „Werkzeuge zur hardwarenahen Programmierung in höheren Programmiersprachen auf 8-Bit-Computern".



Auf der **4. Umschlagseite** stellen wir Ihnen den VIDEOTON-Rechner VT 32 vor. Er ist ein 16-Bit-Mikrorechner für CAD-Aufgaben mit dem UNIX-kompatiblen Betriebssystem DMOS.

Vorschau

Im Heft 12/1988 finden Sie unter anderem Beiträge zu folgenden Themen:
- MS-DOS
- Festplattenorganisation
- ROLANET 1 mit Lichtwellenleitern
- Leipziger Herbstmesse '88

Inhalt



Info

Erste Muster des 1-MBit-RAM übergeben

Die ersten in der DDR hergestellten 1-MBit-Speicherschaltkreise wurden am 12. September 1988 dem Generalsekretär des ZK der SED, Erich Honecker, von einem Kollektiv des VEB Kombinat Carl Zeiss JENA vorfristig übergeben. Diese Leistung konnte nach nur zwei Jahren Entwicklungszeit von den Werktätigen des VEB Forschungszentrum Mikroelektronik Dresden in enger Kooperation mit Einrichtungen der Akademie der Wissenschaften der DDR und des Hochschulwesens erreicht werden.
Damit sind die Grundlagen geschaffen worden, schrittweise die Massenproduktion dieser Schaltkreise vorzubereiten. Wie der Generaldirektor des Kombinates, Prof. Dr. Wolfgang Biermann, erklärte, werden bis dahin noch ein bis zwei Jahre Zeit benötigt, in der konzentriert weitere Investitionen vorgenommen werden müssen.
Der Projektleiter, Prof. Dr. Bernd Junghans, hob als besonders bemerkenswert hervor, daß man sich generell auf Technik aus der DDR stützen konnte. Das treffe auch für die Reinstraumtechnik zu. Aus dem VEB Spurenmetalle Freiberg kommt der neue 125-mm-Wafer, der 90 Chips des 1-MBit-Schaltkreises beinhaltet.
Die ersten Muster des U 61000 D haben inzwischen im Kombinat Robotron, dem Hauptanwender von Speicherschaltkreisen dieser Leistungsklasse, ihre ersten Funktionstests bestanden.
Parallel zu den Arbeiten am 1-MBit-Speicher wird bereits am nächsten Technologieniveau gearbeitet. *ADN*

Zusammenarbeit DDR – Polen beraten

Der Minister für Elektrotechnik und Elektronik der DDR, Felix Meier, und der Minister für Industrie der Volksrepublik Polen, Jerzy Bilip, haben im August in Warschau über die Erweiterung der wissenschaftlich-technischen und ökonomischen Zusammenarbeit, vorrangig bei mikroelektronischen Bauelementen sowie in der Rechen- und Automatisierungstechnik, beraten. Während der Begegnung, die zugleich der Vorbereitung der 22. Tagung des Wirtschaftsausschusses DDR – VR Polen diente, stellten die Partner fest, daß sich die Tätigkeit ständiger Arbeitsgruppen bewährt hat und vereinbarten neue Zielstellungen zur Erhöhung des wissenschaftlich-technischen Niveaus der Erzeugnisse und ihrer effektiven Herstellung. Wichtige Gebiete sind automatisierte Montagetechnologien und die Entwicklung technologischer Spezialausrüstungen, einschließlich Meßtechnik.
Die Seiten unterstrichen, daß sich der Warenaustausch auf dem Gebiet der Mikroelektronik entsprechend einer im März 1987 abgeschlossenen Kooperationsvereinbarung seitdem verdreifacht hat. *ADN*

Anti-COCOM-Liste

Das japanische Ministerium für internationalen Handel und Industrie hat im August in Tokio bekanntgegeben, daß es mit Wirkung vom 1. September eine monatliche Liste von Hochtechnologie-Produkten herausgibt, die nach seiner Auffassung nicht länger unter die sogenannten COCOM-Bestimmungen fallen. Die erste derartige Liste enthält nach Aussage eines Sprechers des Ministeriums 19 Produkte – darunter elektronische Rechner, Textverarbeitungsgeräte und Frequenzmesser.
Weiter wurde mitgeteilt, daß das bisherige Verfahren, wonach alle Anträge japanischer Unternehmen für Exporte in Staaten des RGW und in die Volksrepublik China entsprechend den COCOM-Bestimmungen generell vorab geprüft werden mußten, künftig entfällt. *ADN*

Computershop in Budapest

Die Novotrade AG eröffnete in Budapest ein kleines Warenhaus für Rechentechnik, das sich zur Vorführung der neuesten computertechnischen Einrichtungen und zum Verkauf computertechnischer Artikel gleicherweise eignet. Im Salon bieten neben Novotrade bedeutende ungarische Firmen, so Számalk, SZTAKI, MIGÉRT, SZKI, MOM, Softinvest, Volán-Elektronik sowie die Fernmeldetechnische Genossenschaft, ihre Neuheiten an. Unter einem Dach kann der Besucher die Produkte der professionellen Computer-Hersteller finden, aus ungarischer und importierter Software auswählen. *MP*

Bulgarische Computer vorgestellt

Auf der diesjährigen Perscomp Expo '88 in Sofia zeigten im April 22 Betriebe Bulgariens sowie ausländische Firmen ihre neuesten Erzeugnisse der Mikroelektronik und Robotertechnik, insgesamt 121 Exponate. Besonderes Interesse fanden die neuesten Modifikationen der PC-Familie MIK-16 aus der VRB. Für die unter EC 1647, 4701 und 4702 registrierte Familie dieser Personalcomputer liegen bereits zahlreiche Aufträge für Großserien aus RGW-Ländern vor. Starke Beachtung fanden auch die vom Computersystem Robko gesteuerten Industrieroboter, ferner automatische Projektierungssysteme sowie Systeme für die Steuerung von Laborexperimenten und für Leitungsprozesse. Die Ausstellung und dazugehörige wissenschaftliche Veranstaltungen förderten den Erfahrungsaustausch zwischen Wissenschaftlern und Praktikern, Herstellern und Anwendern. *ADN*

Ehrung für Speichertechnik

Der Solid State Circuits Council des Ingenieurverbandes IEEE hat einen BRD-Forscher für seinen „bedeutenden Beitrag zur Entwicklung dynamischer RAM-Speicher" mit dem Jahrespreis 1988 ausgezeichnet. Bereits Anfang der siebziger Jahre war es Dr.-Ing. Karl-Ulrich Stein bei Siemens gelungen, ein Bit mit nur einem Transistor und einem Kondensator sicher zu speichern und so den Platzbedarf auf einem Siliziumchip deutlich zu verringern. Bis dahin waren pro Bitplatz mehrere dieser Bauelementepaare erforderlich.
Stein sieht seinen Beitrag vor allem „in der symmetrischen Anordnung der Bitleitungen mit Kompensations-Speicherzellen und in einem ebenfalls symmetrischen Lese-/Schreibverstärker pro Bitleitung".
Die Eintransistor-Speicherzelle und die symmetrische Anordnung wurden kurz darauf erstmals für 4-KBit-DRAMs verwendet und dann in allen nachfolgenden Speichergenerationen mit 16, 64 und 256 KBit. Auch die Megabitspeicher mit 1 und 4 MBit arbeiten unverändert nach diesem Prinzip. Beim 4-MBit-DRAM für 4194304 Bit reichen die Kondensatoren als Grabenzellen unter den Transistoren vertikal in den Chip hinein („trench"), um zusätzlich Platz zu sparen. *MP*

Computer in der Mongolei

Bis 1990 soll in Ulan-Bator ein Betrieb für die Montage von Personalcomputern aus importierten Bauteilen und Materialien errichtet werden. In den verschiedenen Zweigen der Volkswirtschaft soll der Einsatz von Rechnern schrittweise ausgedehnt werden. Bewährt hat sich diese Technik bereits bei automatisierten Leitungssystemen in Kombinaten, in Forschungsinstituten und statistischen Einrichtungen. Zu den geplanten weiteren Anwendungsgebieten gehören der Außen- und Binnenhandel sowie die Landwirtschaft. *ADN*

USA-Handelsgesetz negativ bewertet

Japans Regierung sowie Industrie und Banken bewerten das im August in Kraft gesetzte neue USA-Handelsgesetz übereinstimmend negativ. Das Kabinett in Tokio brachte in einer Erklärung seine ernsthafte Befürchtung zum Ausdruck, daß die mit dem Gesetz gegebenen Vollmachten für ein verschärftes Agieren gegen ausländische Konkurrenten die Wirtschaftskooperation bilateral wie international stark behindern sowie die Entwicklung der Weltwirtschaft massiv beeinträchtigen könnten.
Ministerpräsident Noboru Takeshita reagierte nach Presseberichten mit Bestürzung. Der Präsident des Unternehmerverbandes Keidanren, Dr. Eishiro Saito, nannte die Bestimmungen unfair und gegen die Interessen Japans gerichtet.
Das neue Gesetz sei nichts anderes als eine juristische Legitimierung des Protektionismus – so der Tenor vieler japanischer Geschäftsleute. Unter dem Vorwand, das USA-Handelsdefizit abzubauen, werden Konkurrenten, die erfolgreicher auf den internationalen Märkten operieren, offen mit Strafe belegt, heißt es in einer Erklärung von Keidanren. Man erinnert sich im fernöstlichen Industriestaat sehr genau, wie die Reagan-Administration 1987 den „Chipkrieg" eröffnete und die Produkte der japanischen Halbleiterindustrie in den USA mit hohen Zöllen belegte. Ungeachtet einer späteren Revidierung entstehen noch heute daraus für Japan zusätzliche Zollbelastungen von jährlich 165 Millionen Dollar.
Nicht vergessen sind in Tokio auch die Auswirkungen der sogenannten Toshiba-Affäre. Unter dem fadenscheinigen Vorwurf, die COCOM-Bestimmungen verletzt zu haben, wurde ein bedeutender japanischer Hochtechnologie-Konkurrent durch einschneidende Sanktionen der USA weitgehend vom amerikanischen Markt verbannt.
Auch Indonesien ist besorgt über das neue USA-Handelsgesetz, das in Widerspruch zu den Interessen der Entwicklungsländer stehe. Diese Auffassung vertrat der indonesische Handelsminister Arifin Siregar bei einem Treffen mit US-Senatoren in Jakarta. Der protektionistische Kurs der USA behindere das ökonomische Wachstum der Entwicklungsländer und vertiefe Schwierigkeiten, die sich aus der Last der Auslandsverschuldung und dem Anwachsen der Arbeitslosigkeit ergeben, wird in einer Erklärung der Industrie- und Handelskammer Indonesiens betont. *ADN*

Sowjetischer PC

Der an der Moskauer Universität entwickelte Personalcomputer Korwet bildet das Grundmodell eines PC für sowjetische Oberschüler.
Der Korwet ist ein 8-Bit-Computer mit einem Farbdisplay. Der Benutzer kann auf dem Bildschirm in 16 Farben beliebige Darstellungen, Grafiken oder Figuren zeichnen. Die Kapazität des Speichers beträgt 360 KByte. Dies macht es möglich, Grafiken hoher Auflösung anzufertigen. In der UdSSR sind bereits rund 5000 Personalcomputer dieses Typs hergestellt worden. *ADN*

Mexiko setzt auf Computer

Von Null auf 60 ist in Mexiko die Zahl der Produktionsbetriebe für Computertechnik im Zeitraum von 1981 bis jetzt gestiegen. 35 davon stellen Personalcomputer und Peripheriegeräte her. Die positive Entwicklung des Bereichs, in dem gegenwärtig 6400 Werktätige arbeiten, war 1981 vom Ministerium für Handel und industrielle Förderung mit einem Programm eingeleitet worden, das technologische Entwicklung, nationale Zusammenarbeit und internationale Wettbewerbsfähigkeit anvisierte. Die Investitionen in diesem Sektor stiegen von 38 Millionen Dollar 1983 auf etwa 58 Millionen in diesem Jahr. Trotz Beteiligung ausländischer Investoren gehört die Mehrzahl der Betriebe mexikanischen Unternehmen.
Auf international hohem Niveau steht nach Meinung von Experten die mexikanische Produktion von Software, während die Herstellung integrierter Schaltkreise als problematisch gilt, da alles dafür Notwendige bisher importiert werden muß. Man rechnet in diesem Jahr in der Computerbranche Mexikos mit einem Produktionsvolumen in Höhe von etwa einer halben Milliarde Dollar, wovon 350 Millionen auf die Herstellung von Personalcomputern entfallen. Allein 1988 soll mexikanische Computertechnik im Wert von 180 Millionen Dollar ins Ausland gehen. Die Hälfte der Exporte erhalten lateinamerikanische Länder, vor allem Venezuela und Ekuador, und die andere Hälfte die USA und europäische Abnehmer. *ADN*

ASIC – eine Revolution?

Prof. Dr. Dietmar Müller
Technische Universität Karl-Marx-Stadt, Sektion Informationstechnik

Quellen von ASIC

Seit dem Entwurf und der Präparation des ersten Bipolar-Transistors im Jahre 1948 sind zwar zirka 40 Jahre vergangen, jedoch begann die „stürmische" Phase der Entwicklung der Mikroelektronik erst vor etwa 15–20 Jahren.[1]
Diese Phase ist nach wie vor charakterisiert durch eine fast jährliche Verdopplung des Integrationsgrades, das heißt, die Zahl der Transistoren auf dem Chip verdoppelt sich. Dies wird einerseits durch Verringerung der Strukturabmessungen und andererseits durch Vergrößerung der Chipfläche erzielt. Bild 1 zeigt diesen Sachverhalt. Gleichzeitig sind markante Vertreter von Speicher- und Prozessorschaltkreisen eingezeichnet, die diese potentiellen Möglichkeiten nutzen. Die Entwicklung im Bild 1 ist als Toleranzbereich angegeben, da zahlreiche Autoren recht unterschiedliche Angaben und Prognosen fixieren (z. B. /2/ /3/). Insbesondere über die Entwicklung nach 1990 gehen die Meinungen der Experten weit auseinander. Während einige eine Grenze bei zirka 10 Millionen Transistoren erwarten, prognostizieren andere ein stetiges Wachstum bis zu Speicherschaltkreisen von über 100 MBit bei dann vorliegenden Strukturabmessungen unter 0,5 μm bei Chipflächen von 80 mm^2 und Nutzung der dritten Dimension. Die Speicherschaltkreise besitzen dabei nach wie vor Schrittmacherfunktionen, sie erreichen alle drei Jahre ein neues Technologieniveau. Prozessor- und Logikschaltkreise folgen dieser Entwicklung mit einem Abstand von bis zu einem Technologieniveau. Sie weisen aber teilweise bedeutend größere Chipflächen auf (Prognose: bis zu 300 mm^2!).

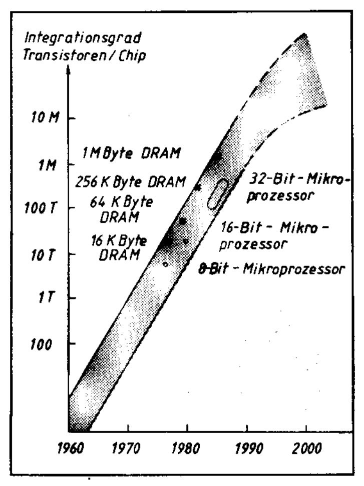


Bild 1 Entwicklung des Integrationsgrades

[1] Selbst der Begriff „Mikroelektronik" war zu dieser Zeit noch nicht festgeschrieben. So formulierte KHAMBATA 1969 /1/:
„Wir stehen am Anfang einer neuen technologischen Ära der Elektronik, deren Ende nicht absehbar ist. Ich kann mir vorstellen, daß sich in einigen Jahren einer meiner Söhne beim Lesen dieses Buches über die lausige Zeit wundert, in der sein Vater sich mit dem alten Phänomen der sogenannten ‚Mikroelektronik' befaßte. Zukünftig wird es vielleicht die ‚Nanoelektronik' oder die ‚Picoelektronik' sein. Wer weiß es?"

Durch diesen ständigen Fortschritt ist eine Veränderung der Hauptrichtung von Forschung und Entwicklung innerhalb der Mikroelektronik eingetreten. In /4/ werden zur Charakterisierung der Zielrichtungen Fragen formuliert, die für die Zeiträume jeweils lauten:
1960–1970: Wie realisieren?
1970–1980: Wie entwerfen?
1980–1990: Was soll entworfen werden?
Während die erste die Technologie, die zweite die Entwurfsmethoden und -verfahren beinhaltet, besitzt die dritte Frage schon fast philosophische Aspekte. Welche komplexe Funktion soll mit 10^7 bis 10^8 Transistorfunktionen pro Schaltkreis realisiert werden, die noch so universell ist, daß eine genügende Einsatzbreite gewährleistet werden kann? Die Antwort auf die obige Frage: „Was soll entworfen werden?" ist bereits bei viel geringeren Transistorzahlen, etwa ab MSI-Niveau mit 100 bis 1000 Transistoren, relevant. Eine Antwort bzw. eine Lösung des obigen Widerspruches sind die ASICs, wobei mehrere Gründe für ihr Entstehen existieren.

Gründe für ASIC

● technische Gründe

Aus den oben genannten Quellen resultieren technische Gründe für das Entstehen von ASIC.
Bisher wurden vom Anwender, das heißt dem Geräte- oder Systementwickler, über einen Systementwurf durch Partitionierung Teilaufgaben formuliert. Sieht man von Lösungen durch Einsatz von Mikrorechnerstrukturen ab, so wurde die Lösung durch den Entwurf von Leiterplatten und deren Bestückung mit TTL- und CMOS-Standardschaltkreisen realisiert. Jedoch auch innerhalb von Mikrorechnerstrukturen besteht neben der Nutzung der bekannten CPU-, Speicher- und Peripherie-Schaltkreise oft die Notwendigkeit, erforderliche Zusatz- und Anpaßlogik in obiger Weise – mit Standardschaltkreisen bestückte Leiterplatte – zu ergänzen.
Diese traditionellen Lösungen besitzen zahlreiche Nachteile, wie:

- großes Volumen
- großer Bauelementeaufwand
- geringe Zuverlässigkeit wegen Vielzahl der Bauelemente und der Lötstellen
- hoher Energiebedarf
- teilweise geringe Geschwindigkeit.

Wenn es gelingt, die Elektronikingenieure, die diese traditionelle Vorgehensweise praktizieren, von der neuen Qualität – dem Entwurf und dem Einsatz von ASIC – überzeugen zu können, so werden die obigen Nachteile vermieden und die nachfolgenden ökonomischen Effekte möglich. Diese Überzeugung wird jedoch nicht immer einfach sein, da der Schaltungsentwurf und insbesondere die bisher praktizierte Schaltungserprobung mittels Laboraufbauten vollständig entfällt, da diese bei ASIC unmöglich ist und durch eine rechnergestützte Simulation, also Nutzung von CAD/CAE-Systemen, ersetzt wird. Dieses neue und ungewohnte Entwerfen setzt neben benutzerfreundlichen CAD-Systemen

ASIC – nur ein neuer Begriff?

Allein durch das Ausschreiben dieser Buchstabenfolge: ASIC = Application Specific Integrated Circuit kann die Frage nicht beantwortet werden.

Hinter ASIC verbergen sich neue quantitative Möglichkeiten der Anwendung der Mikroelektronik. International und national nimmt der Entwurf und der Einsatz von ASICs überdurchschnittlich zu. Es lassen sich bedeutsame ökonomische, zeitliche und technische Vorteile in einer großen Zahl unserer Betriebe und Kombinate erzielen. ASICs stellen *eine* Variante dar, daß die Anwender selbst Beiträge zur Lösung ihrer technischen Aufgabenstellungen mittels Mikroelektronik erbringen.

Die möglichen volkswirtschaftlichen Effekte entstehen jedoch nur in großem Umfang, wenn auch in großem Umfang „alle Elektroniker" der Anwenderindustrie die neuen Möglichkeiten und die Entwurfs- und Einsatzspezifika kennen und diese neuen Möglichkeiten auch praktisch nutzen. Daß diese Breite natürlich auch Kapazitäten im Zyklus 1 und 2 erfordert, ist selbstverständlich.

Dem dafür notwendigen Informieren dient die in diesem Heft beginnende Beitragsfolge. Ziel soll es sein, eine verständliche Einführung in diesen ständig an Umfang und Bedeutung gewinnenden Problemkreis zu geben und, darauf aufbauend, dominierende Vertreter der ASICs, deren Entwurf, deren Einsatz und deren (multivalente) Nutzung darzustellen. Dabei sollen neben internationalen Aussagen und Trends insbesondere die DDR-Entwurfssysteme und damit mögliche Effekte sowie Formen der Zusammenarbeit von Anwendern und Herstellern im Mittelpunkt stehen.

Diese Beitragsfolge stellt einen integralen Bestandteil eines Komplexes von Weiterbildungsmaßnahmen und Informationen dar.

So ist im Fachvorstand Elektrotechnik der KDT innerhalb der Wissenschaftlichen Sektion „Computer- und Mikroprozessortechnik" ein Fachausschuß „ASIC" gegründet worden, der u. a. die Weiterbildungsarbeit im Rahmen der KDT leitet und teilweise durchführt.

Es existieren bereits sporadisch oder periodisch durchgeführte wissenschaftliche Veranstaltungen (z. B. 1. ASIC-Seminar im Territorium Karl-Marx-Stadt – *siehe Seite 325*). Es werden Weiterbildungslehrgänge in verschiedenen Bezirken (z. B. Berlin, Karl-Marx-Stadt) und von verschiedenen Institutionen (Universitäten, Entwurfszentren, Halbleiterindustrie) durchgeführt. Darauf aufbauend und weiterführend werden im Auftrag des Präsidiums der KDT von obigem Fachausschuß eine zentrale Lehrgangskonzeption zum Komplex ASIC vorbereitet und Schulungs- und Informationsmaterial erarbeitet. Auf dieser Basis werden in den Bezirken diese zentralen Lehrgänge ab 1989 durchgeführt werden können.

Neben den Weiterbildungsaktivitäten existieren auch organisierte Formen des Erfahrungsaustausches und Möglichkeiten, konkrete Entwürfe in Anwenderentwurfszentren (z. B. im Schaltkreiszentrum des VEB Textima-Elektronik Karl-Marx-Stadt) durchzuführen bzw. vorhandene (Teil-)Entwürfe nachzunutzen.

Man muß die eingangs gestellte Frage mit einem eindeutigen NEIN beantworten. ASIC ist nicht nur ein neuer Begriff, ASIC ist eine neue Philosophie der Anwendung der Mikroelektronik. ASIC ist eine weitere, aber *notwendige* Nutzungsform der sich ständig weiterentwickelnden Möglichkeiten der Mikroelektronik, die damit auch *notwendigerweise* umfassend genutzt werden kann und muß.

ASIC von A bis Z (C)

ASIC
(Application Specific Integrated Circuit)
Unter ASIC sollten *verschiedene* Arten von Schaltkreisen (PLD, GA, STAZ u. a.) verstanden werden, die für spezielle Anwendungen bei Beachtung von *vorgegebenen Entwurfsregeln* und -restriktionen (vorgegebene Funktionselemente, Grundprinzipien und -strukturen) mit Hilfe *durchgängiger Entwurfssysteme* auch von *potenten Anwendern* entworfen oder programmiert werden.

EPLD
(Erasable Programmable Logic Device)
Programmierbares Logikfeld mit programmierbarer AND-Matrix und fester OR-Matrix (vergleichbar mit PAL), aber verbessertem Konzept – insbesondere Ein- und Ausgabe – zur Realisierung von – im Vergleich mit PLD (siehe dort) – komplexeren Aufgabenstellungen. Sie sind wie EPROM programmierbar.

FC
(Full Custom)
Bezeichnung für einen Voll-Kundenwunsch-Entwurf bzw. -Schaltkreis (VKW). Oft wird formuliert, diese Art wird auf Transistorniveau optimal entworfen (Frage: Unter welchem Aspekt optimal und von wem entworfen?). FCs sollten nur dann zur Klasse der ASICs gezählt werden, wenn sie nicht völlig „frei" von „Profis" entworfen werden.

GA
(Gate Array)
Digitale Grundschaltungen (Gate-Torschaltung) sind in regelmäßiger festgelegter Anordnung (Array-Matrix, Feldanordnung) auf vorgefertigtem Untergrund (Master Slice – teilpräparierte Si-Scheibe, die die standardisierten Transistoren enthält) vorhanden. Damit sind GA-Schaltkreise charakterisiert durch:
– *ausschließliche* Nutzung vorgegebener, standardisierter Grundschaltungen (Funktionselemente, Makros, Bibliothekselemente – z. B. 2fach NAND)
– konstante Anzahl und Lage der Funktionselemente
– konstante Anzahl, Lage und Kapazität der Verdrahtungskanäle
– konstante Pinanzahl und festgelegter Chipfläche, wodurch auch die Gehäuseform fixiert ist.

GE
Gatteräquivalent,
wird als „Maßeinheit" zur Angabe der funktionellen Komplexität von ASICs genutzt.
1 GE = 2fach NAND (= 4 Transistoren bei CMOS-Realisierung)

HKW
Halb-Kundenwunsch-Entwurf bzw. -Schaltkreis,
typische Vertreter: GA und STAZ. Die Wünsche des Kunden können wegen der Vorgabe standardisierter FE nur „halb" erfüllt werden. Die Bezeichnungen VKW, HKW, FC und SC charakterisieren das Typische von ASICs nur ungenügend!

LCA
(Logic Cell Array)
Attraktiver Vertreter der ASICs, vereinigt die Flexibilität und Quantität (bis 5000 GE) von GA-Schaltkreisen mit der Programmierbarkeit – und damit sofortiger Verfügbarkeit – von PLD-Schaltkreisen (siehe dort). Dazu wird eine Matrix programmierbarer (steuerbarer) Logikblöcke (ebenfalls AND-OR-Strukturen) mit programmierbaren Ein-/Ausgangsmakrozellen umgeben.

PAL
(Programmable Array Logic)
Programmierbares Logikfeld (AND-OR-Struktur) mit programmierbarer AND- und fester OR-Matrix.

PLA
(Programmable Logic Array)
Programmierbares Logikfeld mit programmierbarer AND-Matrix und programmierbarer OR-Matrix [Mit Modifikationen auch als FPLA (Field PLA) bezeichnet].

PLD
(Programmable Logic Device)
Einerseits Sammelbegriff für vom Anwender programmierbare ASICs geringer Komplexität (z. B. FPLA, PAL, PLA), andererseits Bezeichnung für eine spezielle Art von ASICs (siehe EPLD).

ROM
(Read Only Memory)
Bekanntes Bauelement in verschiedenen Programmiervarianten als maschinenprogrammierbarer ROM, PROM, EPROM u. a. ROM sind durch eine feste AND-Matrix und eine programmierbare, vollständige OR-Matrix charakterisiert.
PROM kann als Ausgangspunkt für die PLD gelten.

SC
(Semi-Custom)
siehe HKW

SC
(Standard Cell)
siehe STAZ

SK mit allgemeinen Zellen
Diese Zellen sind Funktionsblöcke (Blockmakros, Megazellen) mit großer funktioneller Komplexität (z. B. mit der Funktion eines CPU-Schaltkreises 8086). Im Gegensatz zum Voll-Kundenwunschentwurf sind beim ASIC-Entwurf auch diese allgemeinen Zellen – zwar sehr zahlreich – dennoch Elemente einer Makrobibliothek und damit im allgemeinen vom Anwender nicht entworfen.

STAZ
Standardzellen-Entwurf bzw. -Schaltkreis
Wie bei GA erfolgt Entwurf durch ausschließliche Nutzung vorgegebener, standardisierter Funktionselemente, die jedoch auf der Basis angepaßter, damit unterschiedlich dimensionierter Transistoren entworfen sind. Daher und wegen der variablen Anordnung der Funktionselemente ist keine Vorfertigung möglich.
Damit sind STAZ charakterisiert durch:
– ausschließliche Nutzung vorgegebener standardisierter Funktionselemente
– variable Anzahl und Lage der Funktionselemente
– variable Anzahl, Lage und Kapazität der Verdrahtungskanäle
– variable Pinanzahl und Chipfläche (im Rahmen festgelegter Bondinselringe).

eine Mindestbereitschaft bei den Elektronikingenieuren voraus.

● ökonomische Gründe

Die Triebfeder der ständigen technologischen Weiterentwicklung der Mikroelektronik ist der mit steigendem Integrationsgrad sinkende Preis pro Transistor. Daraus resultiert die wachsende Komplexität auch der bisherigen Standardschaltkreise. Mit diesem Wachsen ergibt sich jedoch der (bekannte) Widerspruch, daß mit dem möglichen Ansteigen der Komplexität der vom Schaltkreis realisierten Funktion auch die Einsatzbreite dieses Schaltkreises und damit seine Stückzahl sinkt. Dies bedeutet, der Vorteil der Mikroelektronik – geringe Preise bei großen Stückzahlen – wird nicht wirksam! Die Lösung dieses Widerspruchs wurde versucht durch konsequente und mehrfache Nutzung der Standardisierung zu erreichen. Zum Beispiel beim Gate-Array-Schaltkreis – als ein dominierender Vertreter der ASICs – durch Standardisierung des Entwurfes und der (Teil-)Standardisierung der Präparation.

[2] In /5/ wurde der Übergang vom Transistor zu standardisierten Grundelementen verglichen mit dem Übergang vom Ziegel zur Großplatte im Bauwesen. Während mit Ziegeln „sowohl Schlösser als auch Katen" mit hohem Aufwand gebaut werden können, erbringt die Nutzung von Wandplatten große Einsparungen an Baukosten und -zeiten bei Beschränkung auf eine spezielle Klasse von Bauwerken.

Natürlich muß jede Standardisierung[2] noch die Möglichkeit einer Individualisierung zulassen. Technisch formuliert: die standardisierten Produkte müssen zur Lösung der individuellen, speziellen technischen Aufgabenstellungen geeignet sein. Bei dem VLSI-Standardschaltkreis „Mikroprozessor" erfolgt dies, indem die universelle (standardisierte) Funktion des Mikroprozessorschaltkreises durch Anlegen von Steuersignal(-folgen) zur individuell notwendigen Funktion überführt wird. Das heißt, der Mikroprozessor wird innerhalb einer Mikrorechnerstruktur durch die Programmierung vom Anwender individualisiert.

Bei den traditionellen Standardschaltkreisen geschieht die „Individualisierung" der technischen Lösung durch gezielte Auswahl und individuelle Verdrahtung der Standardschaltkreise auf der Leiterplatte. Bei einer vereinfachenden Betrachtungsweise kann ein Gate-Array-Schaltkreisentwurf ähnlich, als Auswahl standardisierter Funktionselemente (Bibliothekselemente, Makros) und deren Plazierung und Trassierung auf dem – später realisierten – Chip, aufgefaßt werden.

Der Ersatz einer oder mehrerer Leiterplatten durch einen ASIC erbringt große ökonomische Effekte (Verringerung der Kosten, Erhöhung der Zuverlässigkeit u. v. a.), stellt aber nur eine Art „Einstiegsvariante" dar und muß bei weiteren ASIC-Entwürfen durch Beachtung der systemtechnischen Erfordernisse ergänzt werden.

● systemtechnische Gründe

Diese treten immer mehr in den Mittelpunkt und gewinnen an Bedeutung. Mit anderen Worten: Neue originelle Systemlösungen sind in der erforderlichen Komplexität und den zeitlichen Fristen (Erneuerungsgrad) (fast) nur mittels ASIC realisierbar. Es werden Lösungen möglich, und solche innovativen Lösungen sind auch notwendig, die mit „verfügbaren" Standardschaltkreisen nicht oder die auf der Basis eines beim Schaltkreishersteller in Auftrag gegebenen (Voll-)Kundenwunsch-Schaltkreises oft viel zu spät realisierbar wären.

Wenn der Geräte- oder Systementwickler *seinen* ASIC selbst entwirft, bleibt natürlich auch sein Systemwissen bei ihm und kann so geschützt oder ökonomisch (kostenpflichtige Nachnutzung) verwertet werden.

● rechentechnische Gründe

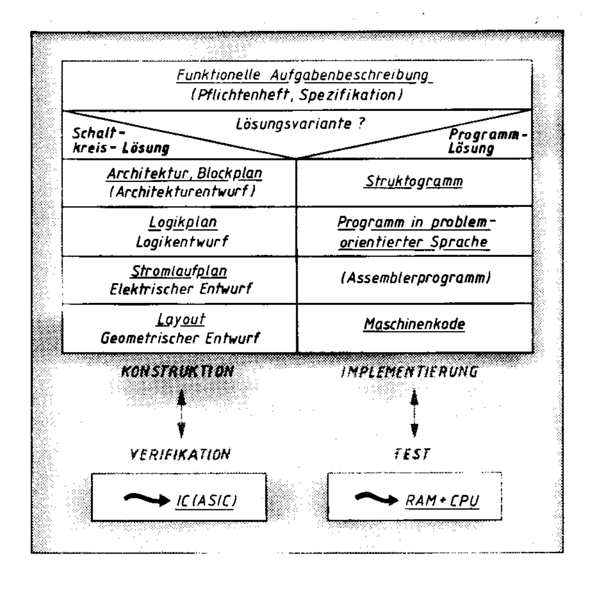
Neben den „herangereiften" Möglichkeiten der Mikroelektronik sind auch die wachsenden und – direkt oder indirekt – am Arbeitsplatz des Elektronikingenieurs verfügbaren rechentechnischen Ressourcen zu nennen. Mit dieser Rechentechnik sind die notwendi-

gen durchgängigen CAD-Systeme, die die „Machbarkeit“ der Entwürfe erlauben, möglich. Diese CAD-Systeme gestatten die Beherrschung der quantitativen Parameter (steigende Zahl der Gatteräquivalente), unterstützen den prüffreundlichen Entwurf sowie den (anzustrebenden) Wegfall eines Redesigns und bieten eine benutzerfreundliche Schnittstelle mit – zukünftig verstärkt – Komponenten des Entwurfs in Ebenen über dem Logikplan und eines wissensbasierten Systems.
Die Benutzerfreundlichkeit entsteht durch Möglichkeiten wie: grafische Ein- und Ausgabe, ausgefeilte Menü- und Windowtechnik, um mit Effektivität die vorhandenen Simulations-, Generierungs- und Hilfsprogramme ohne große rechentechnische Spezialkenntnisse nutzen zu können. Dennoch sind gegenwärtig die angedeuteten Akzeptanzprobleme vorhanden, die aus notwendigem Umdenken – Ersatz von Messungen an „Brettschaltungen“ durch Simulation der Funktion – resultieren. Dieses Umdenken ist vergleichbar mit dem bei Nutzung der Mikrorechentechnik. Im Bild 2 sind vereinfacht, ausgehend von *einer* Aufgabenstellung, zwei unterschiedliche technische Lösungsvarianten und ihre Implementierungs- bzw. Entwurfsebenen skizziert. Die Vergleichbarkeit der Lösungsschritte ist erkennbar. Dabei soll nicht eine Alternative zwischen Mikrorechner- und ASIC-Lösung assoziiert werden, sondern im Gegenteil sollten sich beide sinnvoll ergänzen /6/.

Inhalt von ASIC

Viele neue Begriffe besitzen einen unklaren oder mehrdeutigen Inhalt, so auch der Begriff: ASICs
Der Vorschlag, die integrierten Schaltkreise in zwei Klassen einzuteilen: Standardschaltkreise und Anwendungsspezifische Schaltkreise scheint zu stark vereinfachend zu sein, weil damit auch die Vollkundenwunsch-Schaltkreise, (mit einem möglichen optimalen Entwurf) zur Klasse ASIC gehören würden.
ASIC sollten Schaltkreise sein, die eine
– Klasse von Schaltkreisen mit mehreren Realisierungsformen sind, deren
– Entwurf durch ein leistungsfähiges CAD-System unterstützt wird bei
– Beachtung bestimmter Entwurfsvorgaben (Regeln und Restriktionen) und deren
– Entwurf oder Anpassung auch durch potente Anwender möglich ist.

Bild 2 Vergleichbare Entwurfsebenen für unterschiedliche Realisierungsformen (bei ASIC entfallen elektrischer und geometrischer Entwurf ganz oder teilweise)



ASIC eine Revolution?

Diese Frage muß unter dem Aspekt der Mikroelektronik verneint werden. ASICs sind Ausdruck der kontinuierlich wachsenden quantitativen Möglichkeiten der Mikroelektronik.
Diese Frage kann unter dem Aspekt der Anwendung bejaht werden. ASICs stellen durch ihre Anwendung eine Variante dar, völlig neuartige systemtechnische und ökonomische, eben revolutionierende, Effekte zu erzielen.
Diese Frage kann aber auch unter dem Aspekt des Entwurfs von ASICs bejaht werden. Der Entwurf von ASICs durch die Anwender stellt ebenfalls eine neue, die Breite der Anwendung der Mikroelektronik revolutionierende Variante dar, „... wenn sie die Massen (der Elektronikingenieure) erfaßt ...“.
Daher werden in den nachfolgenden Beiträgen die Spezifika und der Entwurf der verschiedenen ASIC-Arten, die CAD-Systeme und die Schnittstellen zwischen Entwerfer (Anwender) und Hersteller erläutert, um auf diese Weise zur erforderlichen Breite beizutragen.

Literatur

/1/ Khambata, A. J.: Einführung in die Gruppenintegration. Berlin: VEB Verlag Technik 1971
/2/ Broth, J. S. et al.: The Megacell concept an approach to painless custom design. IEE Proc. 1985, No. 2
/3/ Kern, W.; Kind, R.; Merkel, P.; Sickert, K.: Wachstum durch Größtintegration und CAE-Einsatz. Elektronik (1986) 2
/4/ Penfield, P.: Small is Big – The Microelectronics Challenge Memo with in M.I.T., Cambridge 1981
/5/ Garbrecht, A.: Semicustom – IC – Weg zur Wirtschaftlichkeit NTZ 27 (1983) 3, S. 156
/6/ Müller, D.: Gate-Array-Schaltkreise – eine Alternative zu Mikroprozessorschaltkreisen? Wissenschaftliche Zeitschrift der TH Karl-Marx-Stadt (1985) 2, S. 245

✉ KONTAKT ☎

Technische Universität Karl-Marx-Stadt, Sektion Informationstechnik, Reichenhainer Straße 70, Karl-Marx-Stadt, 9022; Tel. 561 31 95

1. ASIC-Seminar 1988

Unter dem Thema „Entwurf und Anwendung von ASIC“ veranstaltete die Technische Universität Karl-Marx-Stadt gemeinsam mit dem Schaltkreiszentrum des VEB Kombinat Textima und dem Fachausschuß ASIC der KDT vom 26. bis 28. April 1988 ein wissenschaftliches Seminar, in dem Spezialisten aus Forschung, Lehre und industrieller Praxis Erfahrungen zu Fragen des Entwurfs und der Anwendung kundenspezifischer Schaltkreise austauschten.
Das Seminar setzte die schon langjährige Tradition der Problemseminare „Schaltkreis- und Systementwurf“ fort.
In 19 Vorträgen wurde über Trends im ASIC-Entwurf, Systembeschreibung, Logikentwurf, Multilevel-Simulation, Layoutbearbeitung, Grafikunterstützung und Entwurfserfahrungen an Hand spezieller ASIC-Entwicklungen berichtet.
Prof. Dr. Müller (TUK) und *Fraikin (ZMD)* hoben in ihren Vorträgen die Bedeutung des Systementwurfes für die modernen hochintegrierten ASIC-Lösungen hervor. *Prof. Dr. Monjau (TUK)* ergänzte die vorgestellten Gedanken durch die Vorstellung moderner Hardware-Beschreibungssprachen. Für den Entwurf von Mikroprogrammsteuerungen *(Dr. Feske, AdW)* bzw. den Logikentwurf *(Prof. Dr. Bochmann, Dr. Steinbach, TUK)* wurden Programmsysteme mit interessanten theoretischen Ansätzen vorgestellt. Eine wichtige Entwicklungsrichtung für den ASIC-Entwurf stellt die 16-Bit-Arbeitsplatzcomputertechnik dar. Dazu berichteten *Pauliuk (TUK)* über ein Programmsystem zur Unterstützung der Schaltungseingabe und Logiksimulation (PCGAD) und *Feustel (Robotron)* über die grafische Schaltungseingabe. Gedanken zu diesem letzten Komplex trug auch *Dr. Leimert (Textima)* vor, wobei sich dies auf eine Erweiterung des bisher genutzten Programmsystems Archimedes bezog. *Dr. Weber (HUB)* stellte ein Layoutbearbeitungssystem (HULDA) vor, das auf PC-Technik lauffähig ist.
Erfahrungen aus der Praxis des U 5200-ASIC-Entwurfs wurden durch die Vorträge von *Wiese, Dr. Böhl (ZMD)* und *Dr. Gerlach (Robotron)* vermittelt. Zum Standardzellen-Entwurf sprach *Krusche (FZW)* und zum Entwurf mit dem ISA-System *Neugebauer (HFO)*. Dem Problem der Entwurfsverifizierung wurde der interessante Vortrag von *Dr. Schwarz* zum Einsatz des Multilevel-Simulators KOSIM gewidmet. Vorstellungen zu alternativen Gate-Array-Systemen vermittelten die Vorträge von *Dr. Herrmann (TUK), Dr. Ritter (MEE)* und *Kühlmann (THI)*.
In einer vielbeachteten Veranstaltung zur Demonstration von Entwurfssoftware auf 16-Bit-PC wurden das Logikentwurfssystem XBOOLE (TUK), der Logiksimulator PCGAD für das Gate-Array-System U 5200 (TUK) und das Layoutbearbeitungssystem HULDA (HUB) vorgestellt.
Das nächste ASIC-Seminar findet im April 1989 statt.

Prof. Dr. sc. techn. D. Müller
Vorsitzender des FA „ASIC“

Ein Baum voller Menüs

Dr. Ulrich Müller, Plauen

Einleitung

Die Programmiersprache PLZ hat nur eine vergleichsweise geringe Verbreitung gefunden, obwohl sie eine Reihe von Vorzügen besitzt. Sie fördert den klaren logischen Aufbau und die konsequente Strukturierung der Programme besser als die meisten anderen Programmiersprachen /1/. Sie kann relativ einfach jeder Programmumgebung angepaßt werden, und es können problemlos auch Programmteile in Maschinensprache entwickelt und eingebunden werden, was PLZ z. B. für bestimmte Aufgaben der Prozeßautomation prädestiniert. Eine nützliche Programmiertechnik stellt die Anwendung rekursiver Lösungen dar. Rekursive Lösungen sind meist kompakter und oft leichter zu schreiben und zu verstehen. Wie das im folgenden vorgestellte Beispiel zeigt, ist Rekursion besonders bequem bei rekursiv definierten Datenstrukturen wie Bäumen.

Die sinnvolle und zweckmäßige Gestaltung der Mensch-Maschine-Kommunikation, der Art und Weise, wie sich jemand, der mit einem Computer arbeiten will, mit diesem verständigen kann, stellt eine wichtige Teilaufgabe bei der Entwicklung von Software dar. Fehler und Versäumnisse, die an dieser Stelle vorkommen, mindern später die Brauchbarkeit des Systems meist ganz erheblich. Dabei sind die unterschiedlichsten Gesichtspunkte zu berücksichtigen: Soll der Computer von einem Spezialisten bedient werden, der über umfangreiche Vorkenntnisse auf dem Gebiet der Rechentechnik verfügt? Werden die Bediener Fachleute auf ganz anderen Gebieten sein, von denen man nicht verlangen kann, daß sie sich umfangreiche Kenntnisse zusätzlich aneignen, nur um den Rechner nutzen zu können? Gibt es eine fest vorgegebene Menge von Kommandos, oder muß die Zahl und Struktur der Kommandos eher flexibel gehalten werden? Ist es unter Umständen erforderlich, mit den Kommandos gewisse Parameter zu übergeben? Es liegt auf der Hand, daß so verschiedene Forderungen nicht mit einem einzigen Konzept erfüllt werden können. Komfortable Betriebssysteme müssen eine andere Kommandostruktur verwenden als ein Programm, mit dem z. B. eine konkrete Teilaufgabe im produktionsvorbereitenden Bereich eines Betriebes bearbeitet werden soll oder das eine Maschine steuern soll. Gerade in den letztgenannten Fällen ist eine möglichst unkomplizierte Bedienung wünschenswert. Eine bewährte Methode dafür ist die sogenannte Menütechnik.

Eine Hierarchie von Menüs

Das Wesen der Menütechnik besteht darin, daß dem Bediener, immer wenn der Rechner ein Kommando erwartet, die zum Zeitpunkt möglichen Kommandos in Form einer Tabelle, dem Menü, angeboten werden. Das Menü enthält neben der Liste der Kommandos auch eine knappe Angabe zur Wirkung der Kommandos. Die Kommandos brauchen deshalb nicht sinnfällig zu sein; es können ohne weiteres Einzelzeichen (Buchstaben und Ziffern) als Kommando benutzt werden. Die Bedienung bleibt trotzdem einfach und erfordert keinen großen Lernaufwand.

In vielen Fällen wird nun eine ganze Folge von Kommandos erforderlich sein, um ein bestimmtes Ziel zu erreichen; nach jedem Kommando bietet sich wieder eine Auswahl von neuen, spezielleren Kommandos, bis der Pfad zu Ende verfolgt und die gewünschte Aktion am Rechner ausgelöst wurde. Aus einem übergeordneten Menü werden die jeweils untergeordneten Menüs aufgerufen, und erst auf der niedrigsten Ebene wird die gewünschte Aktion ausgeführt.

Die Menüs bilden ein hierarchisches System, sie sind gewissermaßen in einem Baum angeordnet. Jede Verzweigung bildet ein Menü. Die Verzweigungen liegen auf verschiedenen Niveaus. Der Bediener erreicht, vom Betriebssystem kommend, auf dem der Menübaum gewissermaßen steht, die erste Verzweigung auf dem nullten Niveau. Man könnte das Menü dort Hauptmenü nennen, weil damit die grobe Vorauswahl zur gewünschten Betriebsart vorgenommen wird. Wir wollen dieses Niveau als höchste Ebene bezeichnen, obwohl sie sich im Bild ganz unten befindet (Der Baum müßte also genaugenommen kopfstehen, aber welches Bild stimmt schon hundertprozentig?). Durch jedes gültige Kommando erreicht der Bediener das nächst niedrigere Niveau, bis sozusagen die ausgewählte Astspitze erreicht ist und die eigentliche Aktion vom Rechner ausgeführt wird. Danach kehrt der Rechner wieder auf das nächst höhere Niveau zurück. Daneben muß natürlich auf jeder Ebene die Möglichkeit bestehen, durch ein spezielles Kommando, z. B. durch Betätigen der Taste OFF das jeweils höhere Niveau zu erreichen (wenn z. B. versehentlich ein falsches Kommando gegeben wurde). Man kann sich also ganz frei in unserem Menübaum bewegen und hat dabei stets vor Augen, welche Kommandos im Moment gerade möglich sind.

Dazu ein konstruiertes Beispiel: Ein für eine Meßplatzsteuerung ausgelegter Mikrorechner soll mit umfangreichen Selbsttestfunktionen ausgestattet sein. Soll etwa ein spezieller RAM-Test ausgeführt werden, so wird der Bediener geführt wie in Bild 1 gezeigt.

Ein rekursives Programm

Die Aufgabe, eine Bedienerführung in Menütechnik zu realisieren, stellt einen Programmierer nicht vor allzu große Probleme. Für jedes Menü ist der Menütext auf dem Bildschirm anzuzeigen; dann muß die Tastatur abgefragt und entsprechend dem Abfrageergebnis zu den Menüs des nächst niedrigeren Niveaus verzweigt werden. Dabei kann vorteilhaft die Unterprogrammtechnik angewendet werden, da bei jedem Menü ganz ähnliche Funktionen zu realisieren sind. Ein BASIC-Beispiel mit großer Flexibilität der Menüs demonstriert /2/.

Eine elegantere Lösung stellt allerdings die Anwendung eines rekursiven Programmes dar. Als rekursiv wird ein Programm bezeichnet, das sich selbst aufruft; damit dies möglich ist, müssen die Parameterübergabe an das Programm und die bezüglich des Programms lokalen Variablen in geeigneter Weise organisiert sein (z. B. über den Stack). Die Blockstrukturen (je nach Programmiersprache als Funktion, Prozedur o. ä. bezeichnet) leistungsfähiger höherer Programmiersprachen besitzen diese Eigenschaft. Die Nützlichkeit der Rekursivität mag vielleicht nicht sofort augenfällig sein. Ein schönes Beispiel, das für eine Reihe technischer Anwendungen interessant sein könnte, zeigt /3/.

Auch für unser Menüprogramm ist eine rekursive Struktur sehr vorteilhaft. Auf jedem Menüniveau wird das Menüprogramm aus dem jeweils höheren Niveau aufgerufen (das Hauptmenü aus dem Betriebssystem); als Parameter wird dabei ein Zeiger, der auf einen Kenndatensatz weist, übergeben, der das jeweilige Menü spezifiziert. Das gesamte Programm, das die Menüs realisiert, reduziert sich damit auf ein Minimum, der Umfang ist unabhängig von der Anzahl der Menüs. Die Struktur des Menübaumes wird auf die Zeigerbeziehungen der Kenndatensätze abgebildet.

Die Kenndatensätze besitzen die im Bild 2 dargestellte Struktur (vgl. dazu die Typenvereinbarung für die Kenndatensätze im PLZ-Programm).

Bild 3 zeigt einen Teil der Kenndatensätze für unser Beispiel. Dem eigentlichen Menüprogramm wird nun ein Zeiger auf den jeweiligen Kenndatensatz übergeben. Dies beginnt in der Hauptprozedur mit dem Zeiger auf den Kenndatensatz (siehe PLZ-Programm Bild 4) des Hauptmenüs. Die niedrigeren Niveaus werden vom Menüprogramm aus aufgerufen. Dazu wird dem Kenndatensatz zunächst der Zeiger auf den Menütext entnommen und einer Prozedur zum Anzeigen von Texten übergeben (die hier wohl nicht näher aufge-

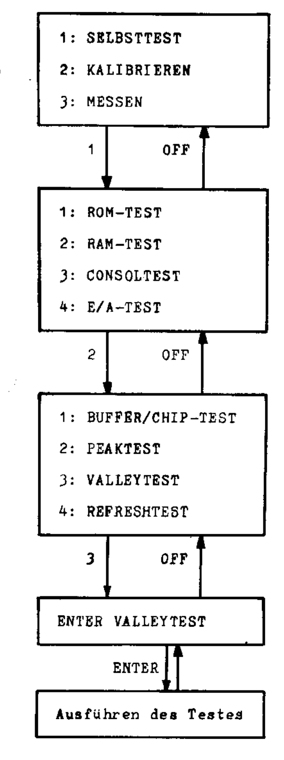


Bild 1 Kommandofolge und Menüs, die zum Valleytest führen

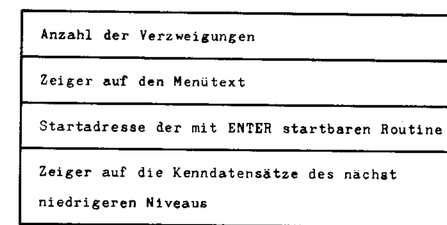


Bild 2 Struktur der Kenndatensätze

```
        global    xec, 10_id
        external  valley
cls:    equ       0ch        ;Steuerzeichen Bild Loeschen
nl:     equ       0dh        ;new line
                             ;Sprungverteiler auf die als
                             ;Parameter uebergebne Adresse
xec:    pop       de         ;Rueckkehradresse
        pop       hl         ;Zieladresse
        push      de
        jp        (hl)
noop:   ret
10_id:
10:     defb      3          ;Anzahl der Verzweigungen
        defw      t0         ;Zeiger auf Menuetext
        defw      noop       ;Startadresse der mit ENTER
                             ;startbaren Routine
        defw      111        ;Zeiger auf Kenndatensaetze
        defw      112        ;des naechst niedrigeren
        defw      113        ;niveaus
111:    defb      4          ;id.anzahl
        defw      t11        ;id.txt_zeig
        defw      noop       ;id.do_adr
        defw      1211       ;id.id_ptr[0]
        defw      1212       ;id.id_ptr[1]
        defw      1213       ;id.id_ptr[2]
        defw      1214       ;id.id_ptr[3]

212:    defb      4
        defw      t212
        defw      noop
        defw      13121
        defw      13122
        defw      13123
        defw      13124
13123:  defb      0          ;hier ist die "Astspitze" erreicht
        defw      t3123      ;es gibt kein Menue mehr
        defw      valley     ;die Aktion wird mit ENTER gestartet
t11:    defb      cls        ;Menuetext
        defm      '1: ROM-TEST'
        defb      nl.
        defm      '2: RAM-TEST'
        defb      nl
        defm      '3: CONSOLTEST'
        defb      nl
        defm      '4:, EA-TEST/'
```

```
onstant
          OFF   :=%16            !Tastencode OFF oder ET2!
          ENTER:=%06             !Tastencode ENTER oder ET1!
type
          wp    ^word
          bp    ^byte
          !Typvereinbarung fuer Kenndatensatz!
          id_tab record [anzahl byte, txt_zeig.bp, do_adr word,
                         id_ptr array [9 wp]]
          idp  ^id_tab
external  !Externals vom Assembler-Tabellenteil!
          xec procedure (do_adr word)
          10_id id_tab          !Kenndatensatz des Hauptmenues!
global
          menu procedure(act_id idp)
               local      c  byte
               entry
               DO
                    print_txt(act_id^.txt_zeig)    !Anzeigen des
                                                    Menuetextes!
                    c :=getchar(CONIN)
                    IF  c
                    CASE OFF THEN RETURN
                    CASE ENTER THEN
                         xec(act_id^.do_adr)
                         RETURN
                    ELSE IF c>'0' ANDIF c<='0'+act_id^.anzahl
                         THEN menu(idp act_id^.id=ptr[c-'1'])

                    FI   FI
               OD
               end menu
          main procedure
               entry
               menu(#10_id)
               end main
```

Bild 4 Auszug aus dem PLZ-Programm

◀ ***Bild 3 Auszug aus dem Tabellenteil, der Kenndatensätze und Menütexte enthält sowie die Prozedur xec***

führt zu werden braucht), so daß der Menütext auf dem Bildschirm erscheint. Danach wird auf der Konsole ein Zeichen erwartet. Je nachdem, welches Zeichen nun eintrifft, führt das Menüprogramm verschiedene Operationen aus: Bei OFF wird das Menüprogramm mit RETURN verlassen, das heißt, es wird in das Menü des nächst höheren Niveaus zurückgekehrt (bzw. vom Hauptmenü ins Betriebssystem).

Wird die Taste ENTER betätigt, so ruft das Menüprogramm die Prozedur *xec*, der es als Parameter die Startadresse für die abzuarbeitende Routine übergibt. Die Prozedur *xec* führt einen Sprung auf die übergebene Adresse aus und untergräbt damit in gewissem Sinne ein Anliegen der Programmiersprache PLZ, nämlich durch eindeutige Sprachdefinition, klaren logischen Aufbau der Programme und Strukturierbarkeit, gute Programmiertechniken zu fördern, weshalb bei PLZ ganz bewußt auf eine Sprunganweisung verzichtet wurde. Andererseits eröffnet die Prozedur *xec* die sonst nicht gegebene Möglichkeit, mit einer Startadressentabelle zu arbeiten. Normalerweise wird nur vom niedrigsten Niveau aus eine solche Routine aufgerufen; auf den höheren Niveaus wird dann eine *noop*-Routine eingesetzt, so daß die ENTER-Taste wirkungslos bleibt (ebenso wie andere ungültige Kommandos). Natürlich könnte an dieser Stelle auch ein Fehlertext ausgegeben werden, der das ungültige Kommando zurückweist.

Zum Anwählen eines gewünschten Zweiges eines Menüs werden im Beispiel die Zahlen 1...n [n = Anzahl der Verzweigungen vom aktuellen Knoten (Menü)] verwendet. Bei Eintreffen eines solchen Kommandos wird das Menüprogramm erneut aufgerufen und mit dem Zeiger auf den neuen Kenndatensatz versorgt, der aus dem übergeordneten Kenndatensatz entnommen wird. Es ist aber auch ohne weiteres möglich, mit jedem Menü eine Tabelle der jeweils gültigen Kommandos zu verbinden oder die Anwahl der Zweige unter Kursorsteuerung vorzunehmen.

Die Programmiersprache PLZ erlaubt eine gute Strukturierung der Daten und einen bequemen Zugriff auf die Elemente des Kenndatensatzes über einen Zeiger auf den Kenndatensatz; das Menüprogramm läßt sich deshalb sehr kompakt und übersichtlich formulieren. Eine Übertragung auf andere Programmiersprachen, die in ähnlicher Weise die Strukturierung von Programmen und Daten unterstützen, ist für anders gelagerte Anwendungsfälle sicher problemlos möglich.

Literatur

/1/ Anwenderdokumentation PLZ/SYS für das Betriebssystem UDOS 1526. VEB Robotron, Karl-Marx-Stadt 1984

/2/ Feichtinger, H.: Ein kleines Expertensystem. mc (1985) 10, S. 107

/3/ Boardman, J. T.; Robson, G. C. A.: Towards a problem-solving methodology for coping with increasing complexity: an engineering approach. The Computer Journal **29** (1986) **2**, S. 161

✉ KONTAKT ☏

VEB Forschung und Rationalisierung, Eugen-Fritsche-Str. 6, PSF 654, Plauen, 9900; Tel. 22778

zur 2. Umschlagseite

3-D-Simulation – interaktiver Entwurf von räumlichen Modellen

Andreas Thierbach
Büro für Stadtplanung beim Rat der Stadt Frankfurt (Oder)

Die Leistungsfähigkeit von Computern beweist sich immer mehr auch in der Anwendung für Systemmodellierungen, Simulationen oder Prozeßoptimierungen, dort, wo eine Vielzahl von Daten und Kriterien mit hoher Geschwindigkeit verknüpft, manipuliert oder bewertet werden muß.
Von wachsendem Interesse ist dabei die Simulation der räumlichen Umwelt, sei es in der Grafik, der Malerei, im Design, in der Architektur oder in der Planung allgemein. Für diese Aufgaben finden aufgrund von wachsender Verbreitung, höherer Leistungsfähigkeit sowie verbesserter Software zunehmend auch Kleinrechner ihre Anwendung.
3-D-Simulation ist ein bildschirmorientiertes Programm für Kleincomputer (KC 85) zur Unterstützung von Entwurfsprozessen, wozu die Modellierung geometrischer, formgestalterischer oder architektonischer Körper und Objekte, die Komposition flächiger oder räumlicher Strukturen, sowie eine räumlich-perspektivische Kontrolle und Korrektur gehören.

Arbeitsprinzip

Baukasten – Modellbau – Modellsimulation
- Erstellen von „Modellbausteinen" beliebiger Form („monolithisch" oder seriell), die auch weiterhin verformbar bleiben
- Arrangieren, Komponieren der Bausteine zum Modell
- Betrachten, Überprüfen und Verändern des Modells aus beliebiger Position (von außen, innen, aus der Bewegung; siehe Bildserie 1, 2. Umschlagseite).

Modellbegriff

Unter Modell wird die Abstraktion von realen oder gedachten räumlichen Objekten oder Strukturen verstanden. Hier sind das digital gespeicherte Daten von Merkmalen, die zur Visualisierung des Modells notwendig sind.
Das *Kanten- und Drahtmodell* (im Unterschied zum Flächen- und Volumenmodell) wird aus einem oder mehreren Elementen (hier auch als Bausteine oder Körper bezeichnet) zusammengesetzt.
Jedes *Element* wird wiederum durch seine *Eckpunkte*, deren Lage im karthesischen Koordinatenraum sowie durch *verbindende Linien* (Kanten oder Strukturen) definiert.
Durch Transformation der Punktkoordinaten auf eine Bildebene nach mathematischen Regeln der perspektivischen Darstellung und Ausgabe der Verbindungslinien auf Bildschirm, Plotter oder Drucker werden die digitalen Modelldaten als Bild sichtbar. Die Darstellung der *Horizontlinie* oder des Koordinatenkreuzes fixiert das Modell optisch im Raum.

Das Programm bietet aufgrund seiner kompakten Struktur, der Maschinenprogrammierung und der vollen Ausnutzung des Speichers ein umfassendes Funktionsangebot, hohe Rechen- und Darstellungsgeschwindigkeit sowie eine Verarbeitung vieler Daten bei einfacher Bedienung und hohem Komfort.
Das Anwendungsspektrum könnte von der Einführung in den rechnergestützten Entwurf (CAD) bis zur direkten Nutzung für CAD reichen:
– Mathematik, Geometrie: 3-D-Visualisierung von Körpern
– Bildkünstlerische Ausbildung, Zeichenunterricht, Darstellungslehre, Entwerfen allgemein: Training des Raum-Körper-Vorstellungsvermögens, Vermittlung der Regeln der perspektivischen Darstellung, Überprüfen flächiger und räumlicher Kompositionen
– Formgestaltung, Architektur, Stadtplanung: Ausbildung und Praxis.

Nutzung und Handhabung

- Generieren linearer, flächiger oder räumlicher Grundelemente durch Punktdateneingabe oder Zeichnen auf dem Bildschirm
- Vervielfältigen, Sortieren oder Löschen von Elementen, Elementeteilen oder Elementegruppen
- Manipulieren und Arrangieren dieser Elemente, d. h. Größen-, Form- und Lageveränderung durch Verschieben, Spiegeln, Drehen, Zerren, Stauchen, Vergrößern oder Verkleinern
- Anlegen von Elementekatalogen auf Kassette, die beliebig abrufbar und in zu bearbeitende Dateien einfügbar sind
- 4 Darstellungsarten:

→ normale Perspektive: Bildausschnitt wird je nach Betrachteraugpunkt, -zielpunkt und eingestellter Brennweite auf den Bildschirm transformiert
Der Augpunkt kann auch im Objekt liegen!
→ totale Perspektive: Angewählter Elementebereich wird im Darstellungsfenster vollständig und in maximaler Darstellungsgröße abgebildet
→ isometrische Darstellung: analog 2., jedoch Abbildung ohne perspektivische Verkürzung, so daß alle Ansichten und Isometrien (Axiometrien) möglich sind
→ Grundrißdarstellung (Draufsicht): automatisch im Hilfsmenü je nach Wahl des Bildausschnittes
→ Bewegungssequenzen: Hinein-, Hindurch-, Vorbeigehen, Überflug, Untersichten, Schwenken usw. (siehe Bildserie 2, 2. US)

- Manipulation oder Korrektur von Einzelkoordinaten, Elementen oder der Gesamtstruktur in allen Darstellungsarten (auch in der Perspektive), damit sofortige Kontrolle der räumlichen Auswirkung der Veränderung möglich
- Manipulation von Betrachteraug- und -zielpunkt sowie deren Bewegungsdifferenzen im Raum
- Kameratechnik: stufenlose Brennweitenwahl (Froschauge bis Superteleobjektiv – Zoom)
- Menütechnik: Anwahl aller Menüfunktionen durch Betätigen der Taste mit dem ersten Zeichen des Menüwortes
- Fadenkreuztechnik: Anwahl von Koordinaten mittels Fadenkreuz direkt auf dem Bildschirm, gesteuert durch Cursortasten.

Hardware: Kleinrechner KC 85/2 oder KC 85/3 mit oder ohne RAM-Expander
Peripherie: Monitor, Datenrecorder, Drukker, Plotter, Videotechnik
Rechenzeiten: bis zur fertigen 3-D-Darstellung
einfache Objekte zirka 1–5 s
komplexe Objekte zirka 5–15 s
Daten – Dateien: Der Datenspeicher wird vollautomatisch verwaltet und ist volldynamisch.
Datenmenge (Raumpunktanzahl N):
450–500 Punkte ohne RAM-Expander
1350–1500 Punkte mit RAM-Expander (16 KByte)
Gesamtspeicherverbrauch: N * 16 Byte
Alle Daten sind abruf- und manipulierbar.

Die Hauptfunktionen:
→ PERS: Perspektivmodus, Bewegungssequenzen
→ HELP: Hilfsfunktionen, Generierungsmodus:
  → WAHL: Wahl eines XY-Koordinatenausschnittes
  → +,–: Vergrößern, Verkleinern des Ausschnittes
  → SUCH: Suchen eines Punktebereiches und Darstellung
  → V1:1: Verhältnis 1:1-Darstellung eines Ausschnittes
  → AUGE, ZIEL: Einstellung mittels Fadenkreuz
  → NPIX, DEFE, ELIN, MANI: siehe Unterfunktionen
  → CORP: Korrektur von Punktkoordinaten mit Fadenkreuz
  → FIND: Finden (Anzeige) der Elementenummern
  → GENE: Generierungshilfe für Elemente: Polygon- und Bogenfunktion, Flächen- und Körperdefinition
→ KOIN: Koordinateneingabe
Korrigieren, Kopieren, Einfügen, Löschen von Punkten

Die Unterfunktionen:
→ WAHL: Wahl der Darstellungsart
→ AUGE: Einstellung Augpunktkoordinaten und -differenz
→ ZIEL: Einstellung Zielpunktkoordinaten und -differenz
→ NPIX: Wahl des aktuellen Punkte- oder Elementebereiches
→ DEFE: Elementedefinition
→ ELIN: Elementeeingabe: Korrigieren, Kopieren, Löschen
→ MANI: Manipulation: Verschieben, Spiegeln, Drehen, Maßverändern, Zerren

Die Nebenfunktionen:
→ FPIX: Einstellung Pixelfenster-Begrenzung:
→ COLO: Einstellung Linien- und Hintergrundfarben:
→ TAFU: Anzeige der wichtigsten Tastenfunktionen

→ > L/S: Datei laden, retten, verknüpfen, hardcopy

Literatur

/1/ Thomae, R.: Perspektive und Axonometrie, Verlag Kohlhammer 1976

/2/ Lampe, B.: Algorithmen der Mikrorechentechnik, Maschinenprogrammierung und Interpretertechniken des U 880 D. 2. Auflage 1983

/3/ Kleincomputer KC 85/2, Systembeschreibung, VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen. Stand 9/85

/4/ Befehlsbeschreibung U 880 D. VEB Mikroelektronik „Karl Marx" Erfurt

/5/ Schlenzig, K. u. S.: Tips und Tricks für kleine Computer. Berlin: Militärverlag 1988

✉ KONTAKT ☏

Büro für Stadtplanung beim Rat der Stadt Frankfurt (Oder), Rathaus, Frankfurt (Oder), 1200; Tel. 365306

EPROMs hoher Speicherkapazität

Jürgen Wrenzitzki,
Ingenieurbetrieb für die Anwendung der Mikroelektronik Erfurt

Die technische Entwicklung im Bereich der Mikrorechner hat in den letzten Jahren gewaltige Fortschritte gemacht. Die Leistungsfähigkeit von 16- bzw. 32-Bit-Mikrorechnern erreicht in bestimmten Bereichen schon heute die Parameter von Minirechnern. Der Trend zur weiteren Miniaturisierung und zur Erhöhung der Arbeitsgeschwindigkeit ist anhaltend. System-Software wird nicht mehr ausschließlich auf Assemblerebene geschrieben. Durch Nutzung von Hochsprachen bei der Systemprogrammierung wird eine relativ leichte Übertragbarkeit von Betriebssystemen auf unterschiedliche Hardwarekonfigurationen ermöglicht. Der Nachteil der umfangreicheren Software wird durch mehr Speichervolumen kompensiert. Neben den RAM-Bausteinen, die heute in Größen bis zu 4MBit produziert werden, haben die EPROMs eine ebensolche Entwicklung erfahren. Für die Anwendung von EPROMs ergeben sich, besonders hinsichtlich der Programmierung dieser Bausteine, neue Gesichtspunkte, die im folgenden näher erörtert werden sollen.

Programmieralgorithmen

Mit dem Anwachsen des Speichervolumens erhöht sich bei der EPROM-Programmierung die Programmierzeit. Bei Anwendung des bekannten Standardalgorithmus vom 50 ms Programmierimpuls pro Speicheradresse ergeben sich Programmierzeiten, die unvertretbar hoch sind /1/. In den Herstellerunterlagen findet man deshalb nur noch Programmieralgorithmen, sogenannte FPA (fast program algorithm), die eine starke Reduzierung der Programmierzeit bewirken. Prinzipiell ähneln sich die zu benutzenden Programmieralgorithmen bei den verschiedenen Herstellern:

– Bei erhöhter Betriebsspannung V_{cc} und angelegter Programmierspannung V_{pp} wird maximal M-mal versucht, die Information mit einem Programmierimpuls der Länge t_{pw} in die ausgewählte Speicheradresse zu programmieren und sofort wieder zu lesen (verify). Bei erfolgreicher Leseoperation wird die angewählte Adresse mit einem Impuls der Länge N ∗ X überprogrammiert, wobei N ein vom Hersteller angegebener Faktor und X die Anzahl der bisher benötigten Impulse ist (intelligente Programmierung) /1/, /2/.

Neuere Programmieralgorithmen benutzen Überprogrammierimpulse konstanter Länge, oder auf die Überprogrammierung wird völlig verzichtet. Die erforderliche Datensicherheit wird herstellerseitig schon durch das implementierte Technologieniveau oder durch *schärfere* und exakter einzustellende Parameter während des Programmierzyklus garantiert /3/, /4/.

Die einzusetzenden Werte für M, N und t_{pw} variieren wie die von den verschiedenen Herstellern kreierten Bezeichnungen für ihren Programmieralgorithmus. Man findet Namen wie INTELligent Programming, Fast Programming, Snap!Pulse Programming, High Speed Program Mode, Quick Pulse Programming usw. In den meisten Fällen wird mit Programmierimpulslängen $t_{pw} = 1$ ms und einer maximalen Impulszahl M = 25 gearbeitet. Die Betriebsspannung V_{cc} wird während der schnellen Programmierung auf 6 V eingestellt, und der Überprogrammierfaktor ist 3 oder 4 /1/, /2/.

Die Programmierspannung der Bauelemente richtet sich nach der Herstellungstechnologie der EPROMs und kann 12,5 V oder 21 V (25 V für 2716 und 2732) betragen. Für extrem kurze Programmierzeiten, speziell bei CMOS-EPROMs, werden präziser einzustellende Parameter gefordert (Beispiel: SNAP!PULSE PROGRAMMING (Texas Instruments) $t_{pw} = 100\,\mu s$, $V_{cc} = 6{,}25$ V, $V_{pp} = 12{,}75$ V), wobei beispielsweise für einen TMS 27C256 Programmierzeiten von weniger als 4 Sekunden erreicht werden.

Bild 1 zeigt die Programmier-Impuls-Diagramme der EPROMs 2764 (27128), 27256 und 27512 für schnelle Programmierung.

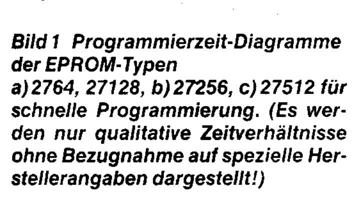
Hinsichtlich der Programmierbarkeit der EPROMs erfolgt eine individuelle Behandlung der einzelnen Speicheradressen. Für schwer programmierbare Adressen bzw. deren schwer programmierbare Einzeldatenbits ergeben sich je nach vorgeschriebenem Programmieralgorithmus Gesamtprogrammierimpulslängen von 75 bis 100 ms pro Byte.

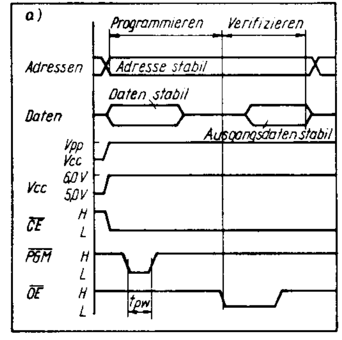
Untersuchungen an einem breiten Typenspektrum verschiedener Hersteller (2764 ... 27256, auch A-Typen und CMOS) zeigen, daß bei einer eingestellten Programmierimpulsbreite von 1 ms (bei $V_{cc} = 6$ V) im allgemeinen nur ein Programmierimpuls pro Byte erforderlich ist.

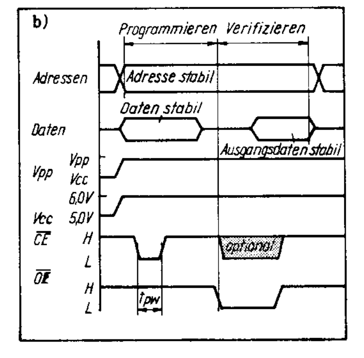
Zur Anwendung des Standardalgorithmus (50 ms/Byte) werden von den Bauelementeherstellern keine Angaben mehr gemacht. Es ist davon auszugehen, daß nur die in den Firmenschriften ausdrücklich angegebenen Kennwerte für einen speziellen EPROM gültig sind. Für Abweichungen von diesen Werten werden durch die Hersteller keine Garantien übernommen, so daß im Einsatzfall kein zuverlässiges Arbeiten des EPROMs erwartet werden muß.

Programmierspannung und Herstellungstechnologie

In zunehmendem Maße liefern die Bauelementehersteller ihre schon traditionellen EPROM-Typen (ab 2764) in scalierten Versionen. Diese sogenannten A-Typen arbeiten mit 12,5 V Programmierspannung (im Gegensatz zu den *traditionellen* 21 V). Manche Hersteller drucken auf ihre EPROM-Gehäuse den Wert der Programmierspannung auf oder fügen ein A an die Typbezeichnung an. Andere ersparen sich diesen *Luxus*, so daß es speziell bei den 32-KByte-EPROMs 27256, die von verschiedenen Herstellern in verschiedenen Technologien gefertigt werden und damit mit unterschiedlichen Spannungen programmiert werden, zu *Programmierproblemen* kommen kann. Fast alle Hersteller bieten ihre EPROMs inzwischen auch als CMOS-Version an. Es ist üblich, ein C in die Typbezeichnung dieser Bauelemente einzufügen. Tafel 1 ist eine Liste von 32-KByte-EPROMs verschiedener Hersteller mit ihren spezifischen Programmierspannungen. Sie gibt außerdem (globale) Auskunft über die Anwendung von Fast-Mode-Algorithmen.







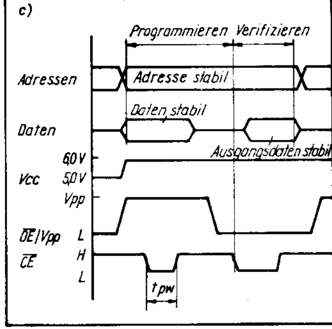


Bild 1 Programmierzeit-Diagramme der EPROM-Typen a) 2764, 27128, b) 27256, c) 27512 für schnelle Programmierung. (Es werden nur qualitative Zeitverhältnisse ohne Bezugnahme auf spezielle Herstellerangaben dargestellt!)

Tafel 1 32-KByte-EPROMs verschiedener Hersteller

Typ	Hersteller	V_{pp} in V	FAST
AM 27256 DC	AMD	12.5	ja
i 27256	Intel	12.5	ja
D 27C256	Intel	12.5	ja
MBM 27256-25	Fujitsu	21.0	ja
D 27256 D	NEC	21.0	ja
D 27256 AD	NEC	12.5	ja
TC 57256 D	Toshiba	21.0	ja
TMM 27256 D	Toshiba	21.5	ja
TMM 27256 AD	Toshiba	12.5	ja
M 27256 FI	SGS	12.5	ja
TMS 27C256	Texas Instruments	12.5	ja*
MSL 27256 K	Mitsubishi	12.5	ja
HN 27256 G-25	Hitachi	12.5	ja

* Bei SNAP!PULSE PROGRAMMING (V_{pp} = 12,75 V; V_{cc} = 6,25 V; t_{pw} = 100 μs) ist die Programmierzeit < 4 s.

Tafel 2 Identifier Codes für Typen und Hersteller

EPROM-Typ	Code (Hex)	Hersteller	Code (Hex)
2732 A	01	Intel	89 (88)*
2764	02	Texas Instr.	97
27C64	07	Toshiba	98
2764 A	08	AMD	01
87C64	37		
27128	83		
27128 A	89		
27256	04		
27C256	8C(04)**		
87C256	80		
27512	0D		
27513	0F		
27011	85		

* 88 bei EPROMs im Plastgehäuse der Firma Intel
** Kennzeichnung nicht eindeutig

Die Einhaltung der Behandlungsvorschriften für Halbleiterbauelemente, speziell der Schutz vor elektrostatischer Zerstörung oder das Fließen von ungewollten Strömen in einer *heißen Fassung*, erhalten besonders bei der Programmierung von CMOS-EPROMs wesentlich mehr Gewicht.
Hinsichtlich der Datensicherheit ist, den Herstellerangaben zufolge, eine mit der Weiterentwicklung der Herstellungstechnologie Hand in Hand gehende positive Entwicklung zu verzeichnen.

Selbst-Identifizierungs-Modus

Neuere EPROMs werden herstellerseitig mit einem Identifizierungs-Code (silicon-signature, identifier mode usw.) versehen, der in zwei Byte sowohl einen Herstelleridentifikator

27513	27512	27256	27128	2764	2732	2716	Pin Nummern (28)	(24)	(24)	(28)	2716	2732	2764	27128	27256	27512	27513
NC.	A15	Vpp	Vpp	Vpp			1			28			Vcc	Vcc	Vcc	Vcc	Vcc
A12	A12	A12	A12	A12			2			27			$\overline{PGM}$	$\overline{PGM}$	A14	A14	$\overline{WE}$
A7	A7	A7	A7	A7	A7	A7	3	1	24	26	Vcc	Vcc	N.C.	A13	A13	A13	A13
A6	A6	A6	A6	A6	A6	A6	4	2	23	25	A8	A8	A8	A8	A8	A8	A8
A5	A5	A5	A5	A5	A5	A5	5	3	22	24	A9	A9	A9	A9	A9	A9	A9
A4	A4	A4	A4	A4	A4	A4	6	4	21	23	Vpp	A11	A11	A11	A11	A11	A11
A3	A3	A3	A3	A3	A3	A3	7	5	20	22	$\overline{OE}$	$\overline{OE}$/Vpp	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$/Vpp	$\overline{OE}$/Vpp
A2	A2	A2	A2	A2	A2	A2	8	6	19	21	A10	A10	A10	A10	A10	A10	A10
A1	A1	A1	A1	A1	A1	A1	9	7	18	20	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$
AØ	AØ	AØ	AØ	AØ	AØ	AØ	10	8	17	19	D7	D7	D7	D7	D7	D7	D7
DØ	DØ	DØ	DØ	DØ	DØ	DØ	11	9	16	18	D6	D6	D6	D6	D6	D6	D6
D1	D1	D1	D1	D1	D1	D1	12	10	15	17	D5	D5	D5	D5	D5	D5	D5
D2	D2	D2	D2	D2	D2	D2	13	11	14	16	D4	D4	D4	D4	D4	D4	D4
GND	GND	GND	GND	GND	GND	GND	14	12	13	15	D3	D3	D3	D3	D3	D3	D3

Bild 2 Übersicht über die EPROMs im 24- und 28poligen Gehäuse

(4Mb)	(2Mb)	(1Mb) 27210	27010	27011	PIN-NUMMERN (40)	(32)	(28)	(28)	(32)	(40)	27011	27010	(1Mb) 27210	(2Mb)	(4Mb)
Vpp	Vpp	Vpp			1					40			Vcc	Vcc	Vcc
$\overline{CE}$	$\overline{CE}$	$\overline{CE}$			2					39			$\overline{PGM}$	$\overline{PGM}$	A17
O15	O15	O15			3					38			NC	A16	A16
O14	O14	O14			4					37			A15	A15	A15
O13	O13	O13	Vpp		5	1			32	36		Vcc	A14	A14	A14
O12	O12	O12	A16		6	2			31	35		$\overline{PGM}$	A13	A13	A13
O11	O11	O11	A15	Vpp/FAST	7	3	1	28	30	34	Vcc	NC	A12	A12	A12
O10	O10	O10	A12	A12	8	4	2	27	29	33	$\overline{PGM}$/$\overline{WE}$	A14	A11	A11	A11
O9	O9	O9	A7	A7	9	5	3	26	28	32	A13	A13	A10	A10	A10
O8	O8	O8	A6	A6	10	6	4	25	27	31	A8	A8	A9	A9	A9
GND	GND	GND	A5	A5	11	7	5	24	26	30	A9	A9	GND	GND	GND
O7	O7	O7	A4	A4	12	8	6	23	25	29	A11	A11	A8	A8	A8
O6	O6	O6	A3	A3	13	9	7	22	24	28	$\overline{OE}$	$\overline{OE}$	A7	A7	A7
O5	O5	O5	A2	A2	14	10	8	21	23	27	A10	A10	A6	A6	A6
O4	O4	O4	A1	A1	15	11	9	20	22	26	$\overline{CE}$	$\overline{CE}$	A5	A5	A5
O3	O3	O3	AØ	AØ	16	12	10	19	21	25	O7	O7	A4	A4	A4
O2	O2	O2	OØ	DØ/OØ	17	13	11	18	20	24	O6	O6	A3	A3	A3
O1	O1	O1	O1	D1/O1	18	14	12	17	19	23	O5	O5	A2	A2	A2
OØ	OØ	OØ	O2	D2/O2	19	15	13	16	18	22	O4	O4	A1	A1	A1
$\overline{OE}$	$\overline{OE}$	$\overline{OE}$	GND	GND	20	16	14	15	17	21	O3	O3	AØ	AØ	AØ

Bild 3 Übersicht über die neuen EPROMs

als auch einen Typidentifikator signalisiert. Diese Kennbytes werden lesbar, nachdem die Adreßleitung A9 auf 12 V angehoben wird und die Adreßleitung A0 geschaltet wird. A0 = Low ergibt den Herstellercode, und A0 = High liefert den Typcode. Die anderen Adreßleitungen können Low oder High sein (Don't care).
Die Anwendung des Identifier-Modes an nicht bekannten EPROMs ist allerdings nicht unbedenklich, da bei einem nicht in diesem Modus arbeitsfähigen Bauelement eine Eingangspegelüberschreitung an A9 erfolgt.
In Tafel 2 sind einige Kennbytes für Hersteller und Typen dargestellt. Bei Anwendung des Identifier-Modes kann die Entscheidung, mit welchem Programmieralgorithmus und bei welcher Programmierspannung zu arbeiten ist, nur aus Herstellercode und Typcode gemeinsam abgeleitet werden. Der weit verbreitete 27256 mit dem Typcode 04 wird von den verschiedenen Herstellern in verschiedenen Technologien produziert und hat unterschiedliche Programmierspannungen. Die sicherste Entscheidungshilfe bleibt in jedem Fall immer noch das Studium der entsprechenden Firmenunterlagen.

Anschlußbelegung und Gehäusebauformen

Neben einer Reihe von EPROMs, die für spezielle Anwendungen entwickelt wurden, hat sich international ein Standard hinsichtlich der Speichergrößen und der Anschlußbelegung durchgesetzt. Diese Bauelemente, die mit einer Datenwortbreite von 8 Bit organisiert sind, werden in Abhängigkeit von ihrer Speichergröße in 24-, 28-, 32- oder 40poligen Gehäusen geliefert. Einige Hersteller produzieren inzwischen auch EPROMs mit einer Datenwortbreite von 16 Bit im 40poligen Gehäuse, die speziell für 16-Bit-Rechner vorgesehen sind.
Bild 2 gibt eine Übersicht über die *Standard*-EPROMs im 24- und 28poligen Gehäuse, und in Bild 3 ist die Anschlußbelegung einiger neuer EPROM-Typen dargestellt, wobei kein Anspruch auf Vollständigkeit erhoben wird. Es wird deutlich, daß die *historisch gewachsene* Anschlußbelegung und auch die Art der Typbezeichnung bis zum 27512 konsequent eingehalten wurde. Bei neuen EPROMs mit mehr als 512 KBit Speichervolumen existiert noch keine einheitliche Bezeichnungsweise, so daß die Anschlußbelegung nicht einfach ableitbar ist. Durch Nutzung von 32poligen Gehäusen werden (bei den 8-Bit-EPROMs) einfach nur mehr Adreßpins zur Verfügung gestellt.
Bei den neuen EPROM-Typen im 28poligen Gehäuse gibt es eine Besonderheit. Sowohl der 27513 als auch der 27011 arbeiten im sogenannten Page-Mode. Durch Doppelnutzung der Datenpins in Verbindung mit anderen Steuersignalen (/we, V_{pp}/RST, /pgm-/we) wird eine seitenweise Adressierung von 16-KByte-Blöcken (entsprechend mehrerer 27128) realisiert. Theoretisch bestünde bei Fortsetzung dieses Prinzips die Möglichkeit, 256 Seiten zu jeweils 16 KByte zu adressieren, was derzeit jedoch technologisch noch nicht realisierbar erscheint.
Im 40poligen Gehäuse werden sowohl EPROM-Chips mit 16 Bit Datenwortbreite (Beispiel: 27210, μPD 27C1024 D) als auch mit 8 Bit Datenwortbreite (Beispiel: μPD 27C1000 D) untergebracht. Aussagen zur Vereinheitlichung hinsichtlich der Anschlußbelegung und der entsprechenden Bezeichnungsweise sind derzeit für diese Gehäusegröße nur bedingt ableitbar.
Es bleibt zu erwähnen, daß die meisten EPROM-Hersteller zu ihren EPROM-Typen sogenannte OTPs (one time programmable) oder Production EPROMs im Plastgehäuse liefern, die nur einmal durch den Anwender programmiert werden können. Für diese Bauelemente machen die Hersteller zudem günstigere Zuverlässigkeitsaussagen. Es gibt außerdem Tendenzen, EPROM-Chips in sogenannte Mini-Flats zu verkappen, was den Anforderungen eines modernen Leiterplattendesigns entgegenkommt.

Literatur
/1/ Chitry, P.; Schramm, M.: Intelligentes Programmieren moderner EPROMs. Radio, Ferns., Elektron. Berlin 36 (1987) 8 S. 498
/2/ Intel MOS-Memories 1984
/3/ Toshiba MOS Memory Products 1985
/4/ Texas Instruments Data Sheet TMS 27C256 1987

⊠ KONTAKT ☏
VEB Mikroelektronik „Karl Marx" Erfurt, Rudolfstraße 47, Erfurt, 5010; Tel.: 621 02/23

EXEC – ein Startprogramm für dBASE II

Christian Hanisch, Berlin

Das Aufrüsten der verfügbaren Standard-Funktionen/Prozeduren einer Programmiersprache durch nutzereigene Unterprogramme in Form problemspezifischer Erweiterungen der „Standard-Funktionen" ist eine für die Anwendungsprogrammierung wirkungsvolle Methode. Assembler-Unterprogramme verbessern den Service und die Leistungsfähigkeit.
Das hier beschriebene Startsystem ist ein Assemblerprogramm, das sich ab der Adresse 0A400H resident macht und dynamisch ein „normales" Programm nachlädt, z. B. dBASE oder REDABAS. In dem resident verbleibenden Teil des EXEC-Programms sind einige Unterprogrammfunktionen enthalten, die z. B. von dBASE durch CALL-Aufrufe genutzt werden können.
Das kopiergeschützte Startprogramm EXEC stellt eine feste Einbindung von Unterprogrammfunktionen in das Startsystem dar. Umfang und Spezifik der implementierten Unterprogrammfunktionen liegen fest. Eine alternative allgemeingültigere Variante ist in Bild 8 und 9 gezeigt, wo einige Funktionen aus EXEC als beliebig einbindbare CMD-Daten herausgezogen worden sind.
Der Umfang und die Anwendungsbeschreibung der realisierten Unterprogrammfunktionen sind aus Bild 1 bis 8 ersichtlich. Die *Funktionen: Bildschirmrollen, Bildschirmsichern/-wiederherstellen (ROLLEN.CMD) sowie Ermittlung des noch freien Speicherplatzes auf Diskette (SPACE.CMD)* sind als separate Moduln herausgezogen und stehen als CMD-Dateien in Form von POKE-Befehlen (erstellt mit dem in /1/ beschriebenen Programm YPOKE) zur Verfügung. Die Aufrufadressen in ROLLEN.CMD und SPACE.CMD sind die gleichen wie unter EXEC. Die Quelltexte in Bild 8 und 9 sind unmittelbar so nutzbar (dBASE II 2.41 und REDABAS 2.0).
EXEC enthält als *?-Funktion* die Dokumentation seiner Anwendungsbeschreibung (Bild 1 bis 7) selbst und erlaubt darüber hinaus mit der *!-Funktion* den Anschluß von mehreren Anwenderdokumentationen oder Bedienungsanleitungen, die mittels ⟨Schirm-Nr.⟩ angezeigt werden; das heißt die *vollständige Anwendungsdokumentation* ist mit in das Startsystem *integrierbar*.

Anmerkung: Obwohl laut dBASE-II-Dokumentation der TPA-Bereich ab 0A400H für Assemblerprogramme verfügbar sein soll, empfiehlt sich eine Nutzung erst ab 0A800H (43008), wobei die BDOS-Beginn-Adresse entsprechend herunterzusetzen ist, um die Assemblerroutinen vor Überschreiben bei Nutzung höherer Versionen (2.43 und 3.0) von dBASE II und REDABAS zu schützen.

```
Rahmenprogramm  E X E C  von  d B A S E-Anwendungs-Systemen

                    V.m=3.2
Aufruf:

A>EXEC[ [<d>:]<filename>[.<type> ¦ .CMD]]<CR>

      <filename>::= Name des  d B A S E -Anfangsprogramms

Ihr dBASE II heisst DBASE ('<CR>')  OR  ('<filename>'):do

Run  ('<CR>') - ('?' ¦ '!') for HELP - OR Cancel ('<<^C>>')?
```

***Bild 1** EXEC-Startprogramm ruft DO.COM auf*

```
Vorliegendes Programm laedt automatisch d B A S E  II
und gibt die Steuerung an ein entweder in der
Aufrufzeile oder laut Anfrage mitgeteiltes Anfangs-
programm    [<d>:]<filename>[.<type>].

Gleichzeitig wird auf die Adresse ØA4ØØh ein integriertes
ASSEMBLER-Unterprogramm  ASSUP  zum Aufruf u. a. mit:
     STORE 41984 TO Madr
     STORE '<from><to>' TO Mparm
     SET  CALL  TO   Madr+6
     CALL  Mparm
zwecks Bildschirmhochrollens im Zeilenbereich von
      <from>::= ØØ .. 22   bis   <to>::= Ø1 .. 23
geladen. (Anm.: z.B. Mparm:='Ø511'; hl -> ¦Ø4¦3Ø¦35¦31¦31¦)
                                          Exit --> ('<ESC>')
GO ON  ('<any key>' ¦ '<Schirm-Nr.>')  -  GO BACK ('<CR>')_
```

***Bild 2** TOOL-Beschreibung Bildschirmrollen*

```
Eine weitere Funktion ist das Sichern und Wiederher-
stellen des Bildschirminhalts:

    NOTE          Sichern <=$=>  Wiederherstellen
    SET CALL TO Madr+9    <=$=>    SET CALL TO Madr+12
    CALL

Die Funktion "Schnelles Bildschirmrollen" (ASSUP+6:=41990)
memory mapped video display ( A5120/30 ) wird ALTERNIEREND
installiert mit:  ( --> Voreingestellt: No memory mapped )

   SET CALL TO Madr+18
   CALL

   Das Datum liegt als  'tt.mm.jj'  auf 42005 (0A415h)
GO ON  ('<any key>' ¦ '<Schirm-Nr.>')  -  GO BACK ('<CR>')_
```

Bild 3 TOOL-Beschreibung Bildschirmsichern/-wiederherstellen

```
Wieviel  Platz in KByte  auf der Arbeitsdiskette
noch frei ist, gibt folgender Aufruf zurueck:
  [STORE 41984 TO Madr]
   STORE 'b' TO Mlw
   STORE   Mlw  + '   '   TO Mparm
   SET CALL TO Madr+15
   CALL Mparm
   Store  Mparm  to Mspace

Die Anzahl der noch freien DIRECTORY-Entries liefert:
   STORE    Mlw  + 'DIR'   TO Mparm
   SET CALL TO Madr+15
   CALL Mparm
   Store  Mparm  to Mfileans
   * Mparm enthaelt rechtsbuendig jeweils das Ergebnis.
GO ON  ('<any key>' ¦ '<Schirm-Nr.>')  -  GO BACK ('<CR>')_
```

Bild 4 TOOL-Beschreibung Freie Diskettenkapazität

```
Zum Schuetzen / Freigeben des Bereiches ab 41984 bis
Beginn des BDOS dient der alternierend wirksame Aufruf:

  [STORE 41984 TO Madr]
   SET CALL TO Madr+3
   CALL

und die Freigabe entsprechend bei jedem zweiten Aufruf.

ACHTUNG: Der Schutz ist  n i c h t  voreingestellt !

Das Schuetzen der  ASSUP-Routinen  ist spaetestens vor der
Nutzung der SORT-Funktion  notwendig.

GO ON  ('<any key>' ¦ '<Schirm-Nr.>')  -  GO BACK ('<CR>')_
```

Bild 5 TOOL-Beschreibung BDOS-Beginn auf 0A400H legen

```
 Das Startsystem geht u.a. fuer den BASIC-Interpreter:

A>EXEC [[<d>:]<filename>[.<typ>]/M:&HA3FF<CR>

"Ihr dBASE II heisst DBASE ('<CR>')  OR  ('<filename>'):"
mit MBASIC<CR> bzw. BASI<CR> oder L80<CR> u.a. beantworten

Beispiel fuer   B i l d s c h i r m r o l l e n:

110     ASSUP2=&HA406:DIM X%(3) ' 3 Integerzahlen <- Mparm
120     X%(1) = &H3004           ' CHR$(4)+"0"
130     X%(2) = &H3135           ' "51"
140     X%(3) = &H8831           ' "1X"  (X unbenutzt)
150     REM  Programm vor dem Rollen ...  und dann:
200     CALL ASSUP2 (X%(1))
GO ON   ('<any key>' ¦ '<Schirm-Nr.>')  -  GO BACK ('<CR>')_
```

Bild 6 TOOL-Beschreibung BASIC-Interpreter unter EXEC
Bild 7 TOOL-Beschreibung TURBO-PASCAL unter EXEC

```
Die Anwendung des Startsystems fuer TURBO-Pascal
ist analog. Um die ASSUP-Routinen zu schuetzen,
muss in O)ptions  E)nd  address: A3FF gesetzt werden.

Angesprochen werden die ASSUP-Routinen ueber:

Procedure ROLLEN (Var fromto: STR4);
 begin     INLINE ( ... ) end;
Procedure SAVEBS; External $A409;
Procedure RETBS;  External $A40C;
Procedure FREEK  (Var Mparm: STR4);
 begin     INLINE ( ... ) end;
Procedure MDATUM (Var Datum: STR8);{0A415h -> 42005}
 begin     INLINE ($2A/DATUM/$11/$15/$A4/$36/$08
                   /$23/$EB/$01/$08/$00/$ED/$B0) end;
GO ON  ('<any key>' ¦ '<Schirm-Nr.>')  -  GO BACK ('<CR>')_
```

```
* ROLLEN.CMD vom: 02.06.87  ( Funktionen nach Bild 2 / 3 )
POKE 41990,195,143,164,195,147,166,195,162,166,201,201;
201,195,35,164,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,33,70,164,237,91,70,165,167,237,82,32,5,33,64;
,164,24,3,33,67,164,17,69,165,1,3,0,237,176,201
POKE 42048,195,72,165.195,70,164,197,42,241,165,22,0,58;
,239,165,95,193,25,16,253,34,141,164,25,237,91,137.164;
,167,237,82,34,139,164,42,141,164,237,91,137,164,167,237;
,82,218,136,164,42,137,164,237,91,246,165,237,75,139.164
POKE 42106,237,176,58,239,165,71,42,141,164,54,32,35,16;
,251,201,0,0,0,0,0,0,229,62,36,50,4,166,33;
,4,166,17,5,166,1,83,0,237,176,225,229,94,35,86:
,42,231,165,167,237,82,225,32,28,229,35,35,94,35,86
POKE 42164,42,233,165,237,82,225,32,14,229,35,94,35.86;
,42,234,165,237,82,225,202,69,165,17,231,165,1,5,0;
,237,176,33,231,165,126,254,4,194,226,165,35,126,230,15;
,71,62,0,184,40,4,198,10,16,252,79,35,126,230,15
POKE 42222,129,71,198,128,50,250,165,58,240,165,61,61,184;
,218,226,165,42,241,165,22,0,58,239,165,95,58,250,165;
,254,128,40,3,25,16,253,34,246,165,237,83,248,165,25;
,34,137,164,33,234,165,126,230,15,71,62,0,184,40,4
POKE 42280,198,10,16,252,79,35,126,230,15,129,254,0,202;
,226,165,71,198,128,50,252,165,58,240,165,61,184,218,226;
,165,195,72,165,237,115,33,164,49,33,164,58,250,165,50;
,2,166,60,71,58,252,165,167,144,218,226,165,60,71,42
POKE 42338,246,165,237,91,248,165,167,25,229,197,6,0,58;
,239,165,79,17,4,166,197,229,65,126,167,32,2,54,32;
,35,16,247,225,193,237,176,33,4,166,6,0,58,239,165;
,79,62,36,237,177,32,7,43,54,164,35,234,144,165,17
POKE 42396,1,166,14,9,205,5,0,58,2,166,60,50,2;
,166,193,225,16,182,42,252,165,34,2,166,58,239,165,71;
,5,5,33,4,166,62,32,119,35,16,252,62,13,119,35;
,62,10,119,17,1,166,14,9,205,5,0,17,251,165,14
POKE 42454,9,205,5,0,17,255,165,14,9,205,5,0,237;
,123,33,164,201,0,0,0,0,0,88,88,88,80,24,0,248;
,88,88,88,0,0,0,0,0,27,129,128,36,24,36;
,27,128,128,0,0,0,0,0,0,0,0,0,0,0
POKE 42512,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0.0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0
POKE 42570,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0
POKE 42628,88,88,88,0,248,128,7,88,88,88,0,0,195;
,215,166,33,237,166,34,21,164,33,0,0,34,142,166,195:
,174,166,33,237,166,34,21,164,33,255,255,34,142,166,0;
,0,0,33,144,166,17,174,166,1,3,0,237,176,237,91
POKE 42686,21,164,237,83,23,164,33,0,0,167,237,82,32;
,7,17,237,166,237,83,23,164,237,83,21,164,237,91.21;
,164,42,135,166,58,142,166,254,255,32,1,235,237,75,137;
,166,237,176,201,201,201,0,0,0,0,0,0,0,0
POKE 42744,,0,0,0,0,0,0,0,0,0,0,0,0,0;
* Fuer Bildschirm: 16 x 64  sind  80,24,0,248 in
* 64,16,0,252 bzw. 0,248,128,7 in 0,252,0,4  zu aendern.
SET CALL TO 41990
RETURN
```

Bild 8 Bildschirmrollen und -sichern/-wiederherstellen

```
* SPACE.CMD vom: 01.06.87          ( Funktion nach Bild 4 )
POKE 41999,229,17,188,165,1,5,0,237,176,58,4,0,50;
,204,165,230,15,14,13,205,5,0,58,188,165,225,254,5;
,210,173,165,254,4,218,173,165,229,42,190,165,17,32,32;
,237,82,225,202,231,164,35,34,202,165,126,203,175,214,65
POKE 42057,230,15,95,14,14,205,5,0,14,31,205,5,0;
,17,7,0,25,78,35,70,3,237,67,195,165,33,0,0;
,34,197,165,34,193,165,17,241,165,14,26,205,5,0,62;
,63,50,209,165,17,209,165,14,17,205,5,0,254,255,40
POKE 42115,67,167,23,23,23,23,23,50,193,165,17,241,165;
,42,193,165,25,126,254,229,40,7,42,197,165,35,34,197;
,165,14,18,205,5,0,254,255,40,30,167,23,23,23,23;
,23,50,193,165,17,241,165,42,193,165,25,126,254,229,40
POKE 42173,226,42,197,165,35,34,197,165,24,217,42,195,165;
,235,42,197,165,235,167,237;82,34,197,165,17,128,0,14;
,26,205,5,0,237,75,197,165,42,202,165,195,82,165,35;
,126,203,175,214,65,230,15,95,34,202,165,14,14,205,5
POKE 42231,0,14,27,205,5,0,229,14,31,205,5,0,35;
,35,126,50,201,165,35,35,35,78,35,70,3,237,67.199;
,165,22,1,225,21,32,4,126,35,22,8,135,48,9,229;
,42,199,165,43,34,199,165,225,11,245,120,254,0,32,12
POKE 42289,121,254,0,32,7,241,237,75,199,165,24,3,241;
,24,215,42,202,165,58,201,165,254,3,40,8,96,105,9;
,68,77,42,202,165,229,33,122,165,80,89,6,5,14,12;
,235,41,235,122,56,4,254,160,56,4,214,160,37,28,13
POKE 42347,32,239,230,240,119,5,40,12,122,230,15,87,35;
,24,224,0,0,0,0,0,225,6,4,14,0,17,125,165;
,26,167,32,8,185,32,5,62,32,119,24,15,203,47,203;
,47,203,47,203,47,230,15,246,48,119,14,1,35,27,120
POKE 42405,254,2,32,2,14,1,16,218,58,204,165,50.4;
,0,230,15,95,14,14,205,5,0,201,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,76,97,98;
,63,63,63,63,63,63,63,63,63,63,63,63,63,63,0
POKE 42463,0,0,0,0,0,0,0,0,0,0,0,0.0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0
POKE 42521,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0
POKE 42579,0,0,0,0,0,0,0,0,0,0,0,0,0;
,0,0,0,0,0,0,0,0,0,0,0,0,0,0
SET CALL TO 41999
RETURN
```

Bild 9 Freie KByte und DIRectory-Entries auf Diskette als dBASE-II-Programm

Literatur

/1/ Hanisch, Ch.: Umwandeln von [Assembler-]COM-Files in dBASE-II-INLINE-POKEs. Mikroprozessortechnik 2 (1988) 10, S. 307

/2/ Bernert, J.; Fischmann, M.; Hahn, W.: Druckersteuerung und Einbinden von Assembler-Programmen bei der Arbeit mit REDABAS. rt/dv 23 (1986) 7, S. 18

/3/ Bernert, J.; Fischmann, M.; Hahn, W.: Die Arbeit mit Textdateien innerhalb des Datenbanksystems REDABAS. rt/dv 23 (1986) 7, S. 20

KONTAKT

Technische Universität Dresden,
Sektion Wasserwesen, Bereich WE;
Tel. 232 61 18

Softwareentwicklung mit FORTH

Lubomir Karadshow, Karsten Noack, Michael Wyschofsky, VEB Werkzeugmaschinenkombinat „7. Oktober", Direktorat Erzeugnisentwicklung

Die Entwicklung der Mikrorechentechnik ist gegenwärtig durch eine Leistungsexplosion der Hardware gekennzeichnet. Das Preis-Leistungs-Verhältnis der angebotenen Systeme verbessert sich ständig. Es ist demnach zu erwarten, daß die Anzahl der Computerinstallationen in den nächsten Jahren sprunghaft ansteigen wird und bedeutende Applikationsbereiche für die Computertechnik erst noch erschlossen werden. Diese Aspekte deuten auf die noch zu lösenden Probleme in der Softwaretechnologie hin. Konventionelle Programmiersysteme werden sich als unzureichend erweisen. Existierende Systementwicklungsmittel sind als Firmen-Know-How im allgemeinen kaum zugänglich und in vielen Fällen nicht hinreichend flexibel /1/.

Entwicklungsmodule auf der Basis der FORTH-Methodik stellen eine Möglichkeit zur Lösung derartiger Problemstellungen dar. Anders als bei herkömmlichen Programmiermitteln liegt hier ein vollständig offenes System vor, von Anfang an in der Sprache realisiert, die es selbst darstellt. Einmalig ist die Tatsache, daß zur Programmentwicklung die gesamte Systemquelle verfügbar ist, und zwar in FORTH. Jede einzelne *systeminterne* Routine ist zugänglich, analysierbar, modifizierbar und für verschiedene Zwecke wiederverwendbar. Der konzeptionell unterstützte Zugriff auf das Gesamtsystem führt zu Eigenschaften, die einmalig sind. Ein Beispiel ist die Möglichkeit, die Funktion bestehender Sprachelemente zu verändern und so fertige Programme ohne wiederholten Entwicklungsaufwand (Edition, Compilation) bzw. Programmabläufe während der Ausführung dynamisch zu modifizieren.

Die neuartige Programmentwicklungsphilosophie, die von FORTH verkörpert wird, zeichnet sich nach außen vor allem durch die sehr einfache Benutzung aller Sprach- und Systemelemente sowie durch die organische Einheit von Systemkern und Applikationserweiterungen aus. Letzteres bedeutet, daß die Hauptbestandteile des Systems (Interpreter, Compiler, Editor, Debugger) in nutzerspezifischer Weise erweiterbar bzw. modifizierbar sind, so daß Applikationen und Systeme mit völlig verschiedenen Eigenschaften und Sprachelementen entstehen können.

Das einfache Programmieren mit eigenhändig definierten Sprachelementen und das unkomplizierte Benutzen eines einheitlichen Entwicklungssystems ermöglichen einen einfachen *Dialog* mit komplizierter Hardware und verkürzen die Test- und Experimentierzeiten beim Programmieren.

Nach Beendigung der Testphase können fertige FORTH-Programme mittels **Metacompilation**, das heißt Regenerierung eines FORTH-Systems durch Compilation von sich selbst /2/, in eine endgültige (optimierte) Version gebracht werden. Mit Hilfe eines sogenannten **Metacompilers** können FORTH-Kernsysteme für andere Mikroprozessoren und Betriebssysteme, Systeme mit modifizierten Eigenschaften sowie optimierte Endcompilate von Programmen erzeugt werden.

Bild 1 Entwicklung mit FORTH-orientierten Systemen

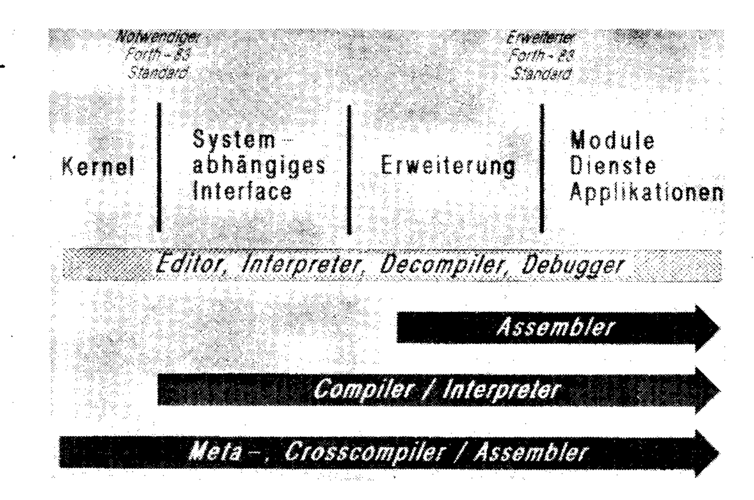


Bild 2 FORTH-83-Programmentwicklungssystem

FORTH-orientierte Systeme sind besonders effektiv für die Entwicklung von (siehe Bild 1):

• Compilern/Interpretern für Spezial- und Fachsprachen

• Programmen für Rechnerarchitekturen, für die keine speziellen Programmentwicklungssysteme vorhanden sind

• Programmen für kleine Rechnersysteme (besonders Einchipmikrorechner)

• fachspezifischen Programmentwicklungssystemen (Programmierung von Maschinensteuerungen)

• Hochleistungssteuerungen auf der Basis von FORTH-Prozessoren (mehrere Millionen FORTH-Operationen pro Sekunde)

• intelligenten Übersetzungssystemen durch Einbindung von KI-Modulen in den Prozeß der Metacompilation.

In den letzten Jahren beobachtet man Anwendung von FORTH in sehr breiten Applikationsbereichen. Insbesondere verstärkt sich der Einsatz in der Bildverarbeitung und der Automatisierungstechnik. Parallel dazu entsteht der Bedarf an Programmentwicklungssystemen und leistungsfähigen Programmierumgebungen, die dem wachsenden Umfang und der steigenden Komplexität der zu lösenden Aufgaben gerecht werden. Ein besonders leistungsfähiges Programmentwicklungssystem wurde im Hinblick auf geplante weiterführende Arbeiten zur Programmierung von Steuerungen mittels moderner Softwaretechnologien (rechnergestützte Übersetzung/Codegenerierung) im Wissenschaftlich-Technischen Zentrum des VEB WMK „7. Oktober" Berlin (NILES), aufbauend auf Ergebnissen der TH Ilmenau (TBK, WB Computertechnik), entwickelt /2/, /3/. Das System ist FORTH-83-kompatibel und vordergründig für 16-Bit-Mikrorechnersysteme konzipiert. Es ist modular organisiert und verfügt über eine Vielzahl von in Applikationen einsetzbaren Erweiterungsmodulen, über einen sehr komfortablen Editor, einen Metacompiler sowie zahlreiche Test- und Dienstprogramme. Es existieren Versionen für die meisten verbreiteten 8- und 16-Bit-Mikrorechner (siehe auch Börse).

FORTH erweist sich immer mehr als eine mächtige und zugleich sehr effektive „Mittlersprache" der hardwaregebundenen und hardwarenahen Programmierung. Zugleich ist FORTH eine Fachsprache zum Bau beliebiger Fachsprachen, die sich durch Objekt- und Prozeßnähe auszeichnen, wie dies für die stark verzweigte Mikroprozessor-Gerätetechnik und eingebettete Mikroprozeßrechner typisch ist. Eine Darstellung des FORTH-83-Programmentwicklungssystems enthält Bild 2.

Literatur

/1/ Goppold, A.: Trends in der Entwicklung der Programmiersysteme. Vierte Dimension 1 (1985) 2, S. 4

/2/ Laxen, H.; Perry, M.: FORTH-83 Model

/3/ Karadjov, L.: Diplomarbeit TH Ilmenau, 1986

/4/ FORTH-SYSTEME für die Industrie. Forth-Systeme, A. Flesch, Titisee-Neustadt

✉ KONTAKT ☎

VEB WMK „7. Oktober", Direktorat Erzeugnisentwicklung/Stammbetrieb, Gehringstr. 39, Berlin, 1120, Tel. 3631641

TURBO-TIP

Anzeige von aktuellem Laufwerk und Pfad

Thomas Koliwer, Jena

In TURBO-PASCAL-Programmen, die auf Rechnern mit MS-DOS-kompatiblen Betriebssystemen (DCP) laufen sollen, ist es manchmal wünschenswert, das aktuelle Laufwerk und den aktuellen Pfad anzeigen zu lassen. Die beiden Funktionen im Bild lösen dieses Problem.

Zur Erläuterung: In der Funktion **aktdrive** wird als Ergebnis das Laufwerk als Großbuchstabe zurückgebracht. Ist ein Kleinbuchstabe erwünscht, ersetze man „$41" durch „$61".

In der Funktion **aktpfad** wird im Register DL das Laufwerk spezifiziert; „0" steht für das aktuelle, „1" für A usw. Nutzer älterer TURBO-PASCAL-Versionen müssen die Strings mit Längenangaben versehen, den Datentyp *Registers* explizit erklären und die „inc(j)"-Anweisung durch „j := j + 1" ersetzen.

In TURBO-PASCAL 4.0 muß die Unit „DOS" eingebunden sein.

```
function  aktdrive:char;
(* bringt aktuelles Laufwerk zurück *)

var reg:  registers;

begin
  reg.ah := $19;
  msdos(reg);
  aktdrive := char(reg.al + $41)
end;


function  aktpfad: string;
(* bringt aktuelles Directory zurück *)

const ad1 : string = '';
        j : byte = 0;

var    reg : registers;
        ad : string;

begin
  with reg do
  begin
    ah := $47;
    dl := 0;
    ds := seg(ad);
    si := ofs(ad)
  end;
  msdos(reg);
  while ad[j] <> #0 do
  begin
    ad1 := ad1 + ad[j];
    inc(j)
  end;
  aktpfad := '\' + ad1
end;
```

Programm für die Fachtagung „Computer- und Mikroprozessortechnik '88" mit internationaler Beteiligung Berlin, 15.–16. Dezember 1988

Plenarvorträge

- Industriecomputer – neue Wege in der Automatisierungstechnik
- Architektur von Hard- und Software der 32-Bit-Klein- und Mikrorechner des SKR
- Der Bildungscomputer robotron A 5105 – ein leistungsfähiger Personalcomputer für das Bildungswesen der DDR
- Kleincomputer in der DDR – Stand und Tendenzen

Sektion A Expertensystem/Mikrorechnersoftware

- Expertensysteme und Fuzzy Reasoning
- Effektive Mikrorechner-Softwaretechnologie auf FORTH-Basis
- Modula-2: Programmiersprache, Werkzeuge und Programmbibliotheken
- Multitask – Programmierabarbeitung im Betriebssystem CP/M-86
- 16-Bit-Echtzeit-Multirechnersystem
- EMR-Entwicklungssystem unter SCPX

Sektion B Rechnerarchitektur/Rechnerhardware

- Aspekte bei Multiprozessorentwicklungen
- Arbeitsweise einer Datenflußarchitektur
- Entwurfsmethodik von Entwicklungs- und Prozeßrechnersystemen
- Ein hochproduktiver Komplex auf Basis eines Prozessors für Matrizen-Rechnungen
- Einbau eines Echtzeit-Zählers und eines Operationsspeichers mit abhängiger Stromversorgung in 16-Bit-Personalcomputer
- Entwurfssystem für VLSI-Speicher

Sektion C Mikrorechner-/Steuerungssysteme

- Modulares Mikroprozeßrechnersystem S 4000
- Projektierung von Mikroprozessor-Moduln für verteilte Steuerungssysteme
- Entwicklung von Software für lokale mikroprozessorgesteuerte dynamische Regler
- Entwurfsmethodik und Sprachunterstützung für verteilte Steuerungen in der Nachrichtentechnik
- Universeller Prozeßkoppelmodul
- PC-Spezialprozessor für binäre speicherprogrammierbare Steuerungen
- Assembler-Interpreter für U 840 PC auf Personalcomputer
- Mikroprozessorsysteme für die Steuerung von technologischen Ausrüstungen
- Bewertung der Produktivität von Mehrprozessor-Steuerungseinrichtungen

Sektion D Testsysteme/Industriecomputerhardware/Logikanalysatoren

- Einführung wissensbasierter Systeme zur Fehlerlokalisierung
- ARGUS – flexibles Testsystem für verteilte Steuerungen
- Paralleltest von peripheren Geräten
- 16-Bit-Prüfrechner-System
- Optimierungsprobleme der Stoßdämpfung der mechanischen Konstruktion von Industriecomputern
- Fragen der mechanischen Konstruktion von Industriecomputern
- Steuerungskonzept eines modularen Logikanalysesystems
- Logikanalysator LA 100
- Der Heimcomputer als Hilfsmittel bei der Übersetzung.

Teilnahmemeldungen an:

Kammer der Technik, Fachverband Elektrotechnik
Clara-Zetkin-Straße 115/117, Berlin, 1086
Telefon: 2265285, Telex: 0114841 techkammer

TERMINE

Jahrestagung der WGMA „Automatisierungstechnik"

WER? Wissenschaftlich-Technische Gesellschaft für Meß- und Automatisierungstechnik der KDT
WANN? 15./16. Dezember 1988
WO? Magdeburg
WAS?
- Organisationsprojektierung
- Büroautomatisierung sowie automatisierte Systeme der Leitung
- Softwaretechnologie, -sicherheit und -zuverlässigkeit

WIE? Teilnahmewünsche bitte an Kammer der Technik, Präsidium, WGMA, Clara-Zetkin-Straße 115-117, Berlin, 1086.

Müller

2. Fachtagung „Computeranimation"

WER? Technische Universität Magdeburg, Kammer der Technik, Institut für Film, Bild und Ton sowie Gesellschaft für den populärwissenschaftlichen Film
WANN? 9. und 10. Februar 1989
WAS?
- Computeranimation für Ausbildung und Training
- Computeranimation und Simulation
- Animationstechnik zur Werbung für Produkte und Verfahren
- Basissoftware für Animation und Simulation

WIE? Vortragsmeldungen (eine halbe Seite Inhaltsangabe sowie Angabe der benötigten Computertechnik) können bis zum 30. 11. 1988 gesandt werden an:
Technische Universität „Otto von Guericke", Sektion Informationstechnik, WB „Methoden der Informatik", PSF 124, Magdeburg, 3010.
Teilnahmewünsche sind bis 15. 12. 1988 zu richten an:
Tagungsbüro der KDT, Brüderstraße 3, Stendal, 3500.

Prof. Dr. Lorenz

PASCAL (Teil 6)

Dr. Claus Kofer
Informatikzentrum des Hochschulwesens an der Technischen Universität Dresden

In dem Maschinenkodeprogramm sind Parameterübergabe- und Aufrufmechanismus des TURBO-PASCAL-Systems nachzubilden. Die Parameterübergabe geschieht über den Stack. Die Parameter werden, beim ersten beginnend, durch PUSH im Stack abgelegt. Der Aufruf erfolgt durch CALL. Vor Verlassen des aufgerufenen Unterprogramms müssen alle Parameter vom Stack entfernt werden. Dabei darf natürlich die zuoberst stehende Rücksprungadresse nicht verlorengehen.
Referenzparameter nehmen im Stack zwei Byte ein. Sie sind die Adresse der entsprechenden Variablen. Records und Arrays werden ebenfalls stets wie Referenzparameter übermittelt. Werteparameter benötigen Platz entsprechend ihres Datentyps.

11.9. Überlagerungsstrukturen

Das TURBO-PASCAL-System bietet die Möglichkeit, *einzelne Prozeduren* des PASCAL-Programms in einer *Überlagerungsstruktur* anzuordnen. Eine solche Prozedur wird durch Voranstellen des reservierten Bezeichners OVERLAY gekennzeichnet.
Prozeduren, die sich überlagern sollen, müssen im Deklarationsteil aufeinanderfolgen. Im Kodebereich des PASCAL-Programms wird für sie nur soviel Speicher vorgesehen, wie für die Aufnahme der größten von ihnen erforderlich ist. Nachfolgend ein Beispiel:

```
PROGRAM Ovl;
PROCEDURE P1;
.
.
.
END;
OVERLAY PROCEDURE P2;
.
.
.
END;
OVERLAY PROCEDURE P3;
.
.
.
END;
PROCEDURE P4;
.
.
.
END;
BEGIN
.
.
.
END.
```

Das Bild 11.6a zeigt die Anordnung der Prozeduren und des Hauptprogramms im Kodebereich. Die Überlagerungsprozeduren P2 und P3 nehmen denselben Speicherbereich ein. Entgegen den Gültigkeitsregeln für Bezeichner kann P2 jedoch nicht durch P3 aufgerufen werden.

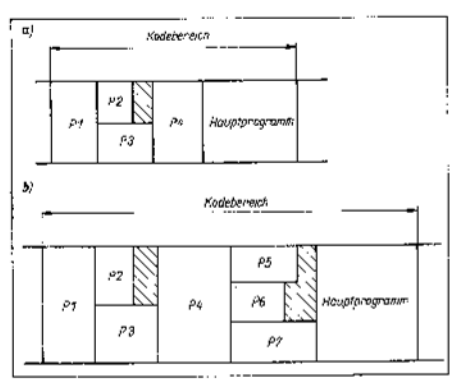


Bild 11.6 Anordnung von Überlagerungsprozeduren im Kodebereich

Auch die Bildung mehrerer Gruppen von Überlagerungsprozeduren ist möglich. Zur Demonstration wird das obige Programm erweitert:

```
...
PROCEDURE P4;
.
.
.
END;
.
.
OVERLAY PROCEDURE P5;
.
.
.
END;
OVERLAY PROCEDURE P6;
.
.
.
END;
OVERLAY PROCEDURE P7;
.
.
.
END;
BEGIN
.
.
.
END.
```

Das Aussehen des Kodebereiches zeigt Bild 11.6b.

Die einzelnen Gruppen von Überlagerungsprozeduren werden als separate Files auf der Diskette abgelegt. Sie bekommen den Namen des entsprechenden COM-Files, aber die Erweiterungen .000, .001 usw.
Das Laden der aufgerufenen Überlagerungsprozeduren organisiert das PASCAL-System. Dabei wird vor dem Diskettenzugriff überprüft, ob sich die gerufene Prozedur bereits im Speicher befindet.

11.10. Modi der Objektkodeerzeugung des Compilers

Beim TURBO-PASCAL-Compiler können drei verschiedene Modi der Objektkodeerzeugung eingestellt werden:
- Hauptspeichermodus
- COM-Filemodus
- Kettenfilemodus.

Im Hauptspeichermodus wird der Objektkode im Hauptspeicher abgelegt. Dieser Modus ist für die Programmentwicklung vorgesehen. Für Produktionsanwendungen ist er nicht geeignet, da zur Abarbeitung des PASCAL-Programms das gesamte TURBO-PASCAL-System erforderlich ist.
Im COM-Filemodus werden COM-Files des Betriebssystems erzeugt. Sie bestehen aus PASCAL-Laufzeitsystem, Programmkode und Programmvariablen. Das COM-File hält die Konventionen des Betriebssystems ein und unterscheidet sich nicht von anderen abarbeitbaren Programmen.
Im Kettenfilemodus werden sogenannte Kettenfiles erzeugt. Sie sind zur unmittelbar aufeinanderfolgenden Ausführung von PASCAL-Programmen vorgesehen. Kettenfiles sind den COM-Files sehr ähnlich. Bei ihnen fehlt lediglich das PASCAL-Laufzeitsystem. Die Abarbeitung eines Kettenfiles kann deshalb nur durch ein anderes PASCAL-Programm eingeleitet werden. Dafür gibt es die Standardprozedur
CHAIN(f).
f muß eine Filevariable sein, der vorher mit ASSIGN das zu startende Kettenfile zugewiesen wurde. Das Kettenfile verdrängt den Kode des rufenden Programms. Mit Hilfe von absoluten Variablen kann es mit Daten versorgt werden.
In gewissem Grade kann der Programmierer darauf Einfluß nehmen, wie sein PASCAL-Programm im Hauptspeicher plaziert wird. Das Bild 11.7 zeigt für die besonders wichtigen COM-Files die hier interessierenden Details. Das gesamte Programm beginnt stets am Anfang des TPA bei der Adresse 100H mit dem etwa 8 KByte großen Laufzeitsystem. An das Laufzeitsystem schließt sich der Kode der Prozeduren und Funktionen sowie des Hauptprogramms an. Standardmäßig beginnt der Kode auf dem ersten freien Byte hinter dem Laufzeitsystem. Es ist aber möglich, die Startadresse zu erhöhen, um

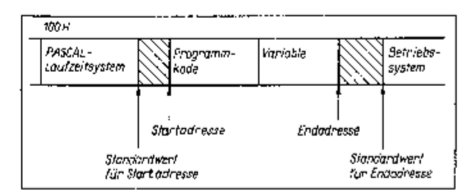


Bild 11.7 Wirkung von Start- und Endadresse auf die Plazierung von PASCAL-Programmen im Hauptspeicher

hier Platz freizulassen für absolute Variablen. Endadresse stellt die höchste für das PASCAL-Programm zu verwendende Adresse dar. Standardmäßig ist sie etwa 700 Byte kleiner als die obere Grenze des TPA. Die Endadresse kann verringert werden, um z. B. Programme zu erzeugen, die auf einem Rechner mit kleinerem TPA laufen können, oder um Platz für Maschinenkodeunterprogramme freizuhalten.

11.11. Quellprogrammuntergliederung

TURBO-PASCAL gestattet mit Hilfe der INCLUDE-Option des Compilers die *Untergliederung* eines PASCAL-*Quellprogramms* in mehrere getrennte Files. Durch sie wird sofort die Quelltexteingabe des Compilers auf das angegebene File umgelegt und dieses bis zum Ende gelesen. Das Bild 11.8 zeigt ein Beispiel.

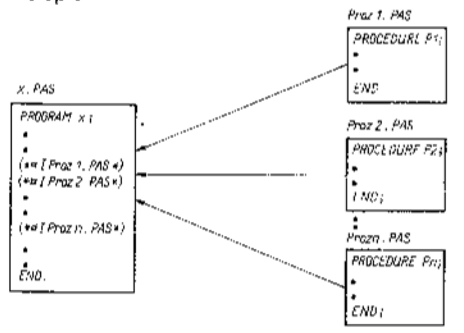


Bild 11.8 Untergliederung eines PASCAL-Programms in mehrere Quelltextfiles

TURBO-PASCAL unterscheidet zwischen *Haupt- und Arbeitsfile*. Der Compiler beginnt die Übersetzung stets mit dem Hauptfile.

Das System hält das Arbeitsfile in einem internen Quelltextbereich, in dem es mit dem Editor korrigiert werden kann.

Haupt- und Arbeitsfile werden zu Beginn der Arbeit vom Nutzer festgelegt. Während das Hauptfile nur durch den Nutzer verändert werden kann, wählt das TURBO-PASCAL-System das File zum neuen Arbeitsfile, bei dem während der Übersetzung ein Fehler auftrat.

11.12. Bedienung des TURBO-PASCAL-Systems

Die Bedienung des TURBO-PASCAL-Systems erfolgt mit Hilfe von einfachen prägnanten Menüs. Kommandoeingaben bestehen aus einem einzigen Buchstaben bzw. aus Filebezeichnungen und Adressen.

Die erlaubten Kommandos werden am Zustandsdiagramm des TURBO-PASCAL-Systems erläutert. Es wird im Bild 11.9 gezeigt.

Nach dem Start geht das System in einen Anfangszustand. Hier kann über das Laden der Fehlertexte des Compilers entschieden werden. Sie beanspruchen einen Platz von etwa 1600 Byte.

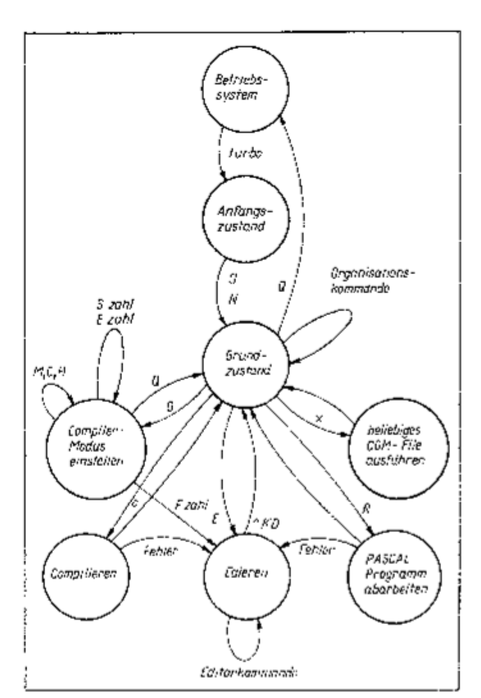


Bild 11.9 Zustandsdiagramm und Bedienerkommandos des TURBO-PASCAL-Systems

Im Grundzustand können zunächst folgende Organisationskommandos eingegeben werden:

- L Wahl des aktuellen Laufwerkes (A:, B:, ...)
- D Anzeige Inhaltsverzeichnisinformationen
- M Angabe des Hauptfiles. Es wird zur Eingabe eines Filenamens aufgefordert.
- W Angabe des Arbeitsfiles. Es wird zur Eingabe eines Filenamens aufgefordert.
- S Abspeichern des Arbeitsfiles auf Diskette.

Durch folgende Kommandos erfolgt der Übergang in die anderen Zustände:

- E Edieren. Im Editorzustand ist die Eingabe von Korrekturkommandos möglich. Es wird die im Anhang 1 angezeigte Teilmenge der TP-Kommandos angeboten. Der Zustand wird durch ^KD verlassen.
- C Compilieren. Es wird das Hauptfile übersetzt. Falls keines angegeben wurde, nimmt der Compiler das Arbeitsfile. Bei Auftreten eines Fehlers wird sofort in den Editorzustand gewechselt und der Cursor auf die Fehlerstelle positioniert.
- R Starten des übersetzten PASCAL-Programms. Falls noch keines übersetzt wurde, wird eine Compilierung eingeschoben. Tritt bei der Programmabarbeitung ein Fehler auf, wird eine entsprechende Meldung ausgegeben, nach Betätigung der ESC-Taste in den Editorzustand gewechselt und der Cursor auf die Fehlerstelle positioniert.
- X Ausführung eines beliebigen COM-Files. Das TURBO-PASCAL-System wird bis auf einen kleinen residenten Teil im Speicher überschrieben. Nach Beendigung der Abarbeitung des COM-Files regeneriert der residente Teil den alten Zustand des TURBO-PASCAL-Systems wieder.
- O Compilermodus einstellen. In diesem Zustand ist die Eingabe weiterer Kommandos möglich:
- M,C,H Einstellung des Hauptspeicher-, COM-File- oder Kettenfilemodus
- Sn Setzen der Startadresse auf n
- En Setzen der Endadresse auf n
- Fn Suchen der Quelltextpassage, zu der die Kodeposition n gehört, und Wechsel in den Editorzustand
- Q Rückkehr in den Grundzustand.

Das Kommando Fn dient dazu, Laufzeitfehler in COM-Files zu lokalisieren.

11.13. Separate Übersetzung

Ab Version 4.0 bietet TURBO-PASCAL die Möglichkeit der separaten Übersetzung von einzelnen Teilen eines PASCAL-Programms unter voller Beibehaltung der Kontrollen des Compilers. Damit wird die Bildung von Programmbibliotheken für oft benutzte Prozeduren und Funktionen ermöglicht sowie die kollektive Erarbeitung von großen Softwareprojekten unterstützt.

Die für die separate Übersetzung erforderlichen Ausdrucksmittel fügen sich organisch in die Sprache ein. Bereits existierende PASCAL-Programme sind ohne Änderungen auch in Version 4.0 lauffähig.

Die Syntaxdiagramme der Bilder 11.10 bis 11.15 zeigen die erforderlichen Spracherweiterungen. Es wird entsprechend Bild 11.10 zunächst eine Übersetzungseinheit eingeführt. Sie ist alternativ das schon bekannte Programm oder eine Unit.

Den Aufbau einer Unit zeigt das Syntaxdiagramm Bild 11.11. Auf den reservierten Bezeichner *UNIT* folgen Unit-Name sowie Interface- und Implementationsteil. Den Aufbau des Interfaceteiles zeigt Bild 11.12. Er wird durch den reservierten Bezeichner *INTERFACE* eingeleitet und besteht aus einer – möglicherweise auch leeren – Folge von Deklarationen von Marken, Konstanten, Typen, Variablen sowie Prozedur- und Funktionsköpfen. Alle im Interfaceteil deklarierten Objekte haben globale Bedeutung. Sie können in einem Programm oder in anderen Units benutzt werden.

Den Aufbau des Implementationsteils zeigt Bild 11.13. Er wird durch den reservierten Bezeichner *IMPLEMENTATION* eingeleitet. Darauf folgt ein Block, der entsprechend Bild 5.1 aus Deklarations- und Anweisungsteil besteht. Seine Deklarationen sind jedoch aufgrund der Gültigkeitsregeln von PASCAL nur innerhalb des Implementationsteils gültig. Enthält der Interfaceteil Prozedur- und Funktionsköpfe, dann muß im Implementationsteil die vollständige Deklaration dieser Prozeduren und Funktionen folgen.

Die Unit *Counter* demonstriert dies:

```
UNIT Counter;
INTERFACE
   PROCEDURE IncCount(amount: INTEGER);
```

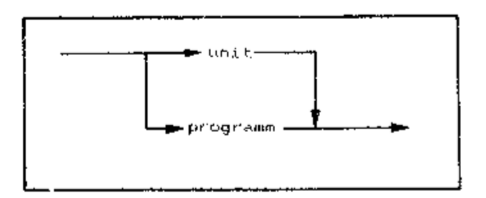


Bild 11.10 Syntaxdiagramm „übersetzungseinheit"

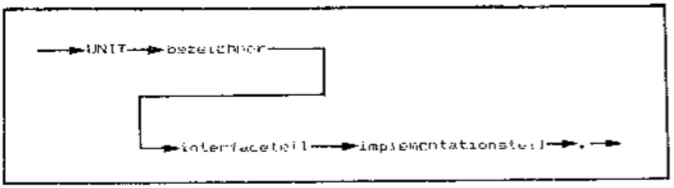


Bild 11.11 Syntaxdiagramm „unit"

Bild 11.13 Syntaxdiagramm „implementationsteil"

IMPLEMENTATION → block →

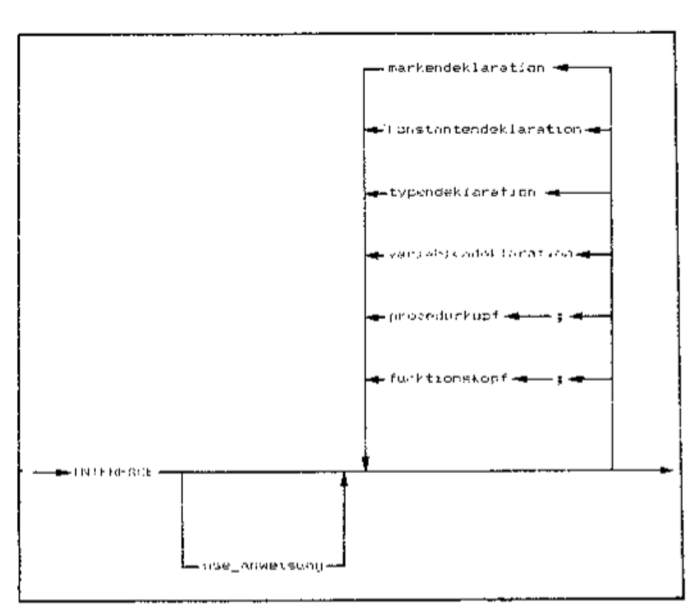


Bild 11.12 Syntaxdiagramm „Interfaceteil"

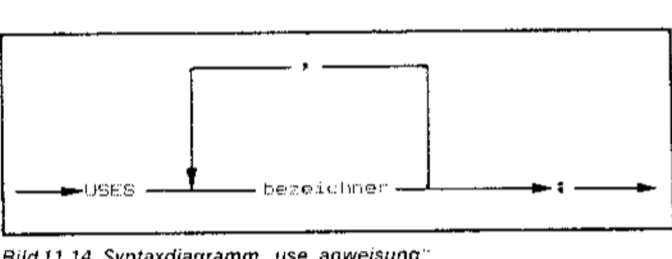
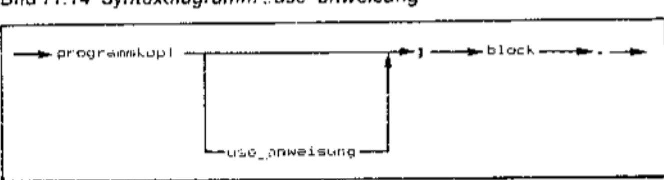


Bild 11.14 Syntaxdiagramm „use anweisung"



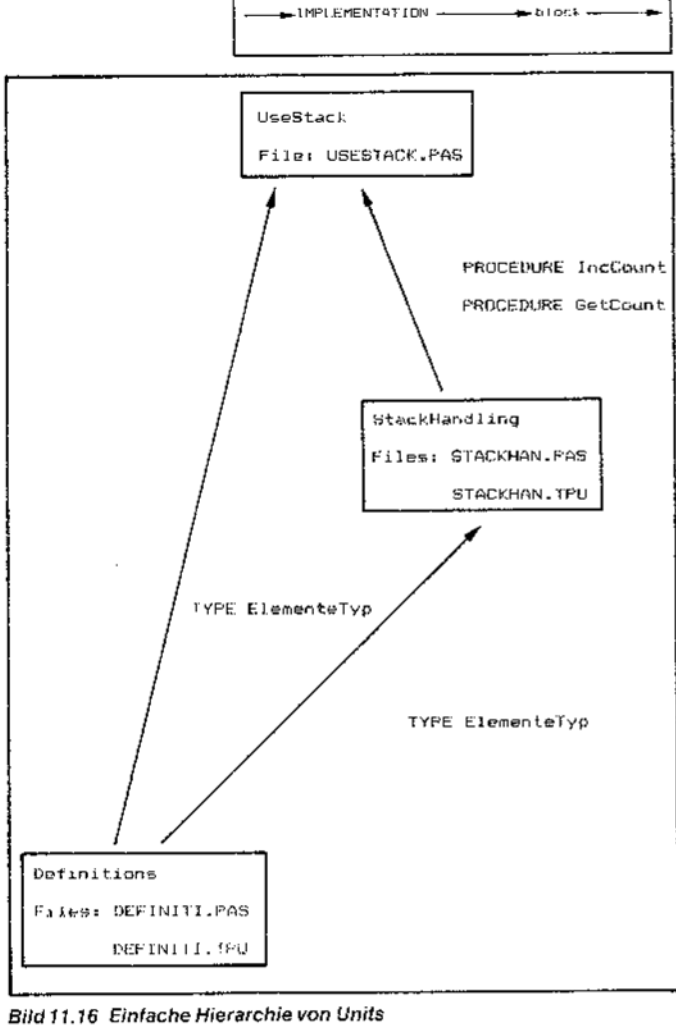


Bild 11.16 Einfache Hierarchie von Units

◀ *Bild 11.15 Modifiziertes Syntaxdiagramm „programm"*

```
  PROCEDURE GetCount(VAR i: INTEGER);
IMPLEMENTATION
  VAR Count: INTEGER;
  PROCEDURE IncCount(amount: INTEGER);
  BEGIN
    Count := Count + amount
  END;
  PROCEDURE GetCount(VAR i: INTEGER);
  BEGIN
    i := Count
  END;
  BEGIN
    Count := 0
  END.
```

Der Interfaceteil der Unit *Counter* wird durch die Köpfe der Prozeduren *IncCount* und *GetCount* gebildet. Dadurch können sie in anderen Übersetzungseinheiten benutzt werden. Der Implementationsteil enthält die Deklaration der Variablen *Count* und die vollständige Deklaration der Prozeduren *IncCount* und *GetCount*. Die Variable *Count* ist außerhalb des Implementationsteils und damit außerhalb der Unit *Counter* nicht bekannt. Der Anweisungsteil BEGIN Count:=0 END des Implementationsteils dient im Programmbeispiel zur Initialisierung der Zählvariablen *Count*. Das TURBO-PASCAL-System sichert ab, daß die Anwendungsteile aller Units vor dem Anweisungsteil des Programms, das die Units benutzt, abgearbeitet werden.

Eine Unit muß nicht notwendigerweise ausführbare Anweisungen enthalten. Im nachfolgenden Programmbeispiel wird durch die Unit *Definitions* lediglich der Datentyp *COMPLEX* deklariert:

```
UNIT Definitions;
INTERFACE
  TYPE
    COMPLEX = RECORD
      re: REAL;
      im: REAL
    END;
IMPLEMENTATION
BEGIN END.
```

Implementations- und Anweisungsteil dieser Unit werden nicht benötigt und sind deshalb leer.

Zur Benutzung der in Units deklarierten Objekte ist die im Bild 11.14 gezeigte Use-Anweisung erforderlich. Sie wird entsprechend Bild 11.15 in das bereits bekannte Syntaxdia-

gramm *Programm* eingefügt. Sie stellt eine Liste von Namen der verwendeten Units dar. Das folgende Programmbeispiel zeigt, wie die Objekte der Unit *Counter* benutzt werden können:

```
PROGRAM demo;
USES Counter;
VAR i: INTEGER;
BEGIN
  IncCount(3); (* . . . *) IncCount(4);
  (* . . . *)
  GetCount(i);Writeln('Zählwert: ',i);
END.
```

Im Programmbeispiel werden die Objekte der Unit *Counter* mit dem Namen referenziert, mit dem sie deklariert wurden. Im Beispiel die Prozeduren *IncCount* und *GetCount*. Es ist jedoch durchaus denkbar, daß das nutzende Programm selbst Objekte gleichen Namens deklariert. Nach den Gültigkeitsregeln von PASCAL überdecken diese dann die Bezeichner der benutzten Units. Würde das Programm Demo z. B. die Anweisungen
USES Counter;
VAR IncCount: BOOLEAN;
enthalten, dann überdeckt die Variable *IncCount* die Prozedur *IncCount* aus der Unit *Counter*. Werden in einem solchen Fall die Objekte der benutzten Units dennoch benötigt, so sind sie mit dem Namen der Unit zu qualifizieren. Das heißt, ihnen ist der durch einen Punkt getrennte Unit-Name voranzustellen. Aufrufe der Prozedur *IncCount* haben dann die Form Counter.IncCount(3).
Es sind einige Erläuterungen zur Implementation der separaten Übersetzung in TURBO-PASCAL zweckmäßig. Units werden durch den Compiler (im Disk-Mode) in ein File mit dem Typ TPU übersetzt und auf der Diskette abgelegt. Für die Übersetzung von Programmeinheiten, die diese Unit benutzen, wird das korrespondierende TPU-File benötigt. Sein Name wird durch den Compiler aus dem Unit-Namen und dem Typ TPU gebildet. Dieser einfache und klare Mechanismus führt zu der Einschränkung, daß der Filename mit dem Unit-Namen übereinstimmen muß. Für das vorangegangene Beispiel heißt dies, daß der Programmtext der Unit *Counter* im File *COUNTER.PAS* abgelegt werden muß. Der Compiler übersetzt es in das File *Counter.TPU*, welches zur Übersetzung von *DEMO.PAS* benötigt wird.
Das Bild 11.12 zeigt, daß im Interfaceteil einer Unit ebenfalls eine Use-Anweisung angegeben werden darf. Dadurch ermöglicht TURBO-PASCAL die hierarchische Strukturierung von Units.
Vom TURBO-PASCAL-System werden dem Anwender eine Reihe von fertigen Units zur Verfügung gestellt, die zum Beispiel zur Arbeit mit dem Bildschirm dienen oder den Aufruf von Betriebssystemleistungen ermöglichen.

Anhang

A. Editorkommandos

A.1 Cursor-Positionierung

Zeichen links	^S
Zeichen rechts	^D
Wort links	^A
Wort rechts	^F
Zeile hoch	^E
Zeile tief	^X
Rollen hoch	^Z
Rollen tief	^W
Blättern hoch	^R Bild zurück
Blättern tief	^C Bild vorwärts
Zeilenanfang	^Q^S
Zeilenende	^Q^D
Bildanfang	^Q^E
Bildende	^Q^X
Fileanfang	^Q^R
Fileende	^Q^C
Blockbeginn	^Q^B
Blockende	^Q^K
Letzte Position	^Q^P Rückkehr zur letzten Cursorposition, vor Suchen, Ersetzen oder Abspeichern

A.2 Einfügen und Löschen

Insert-Modus Ein/Aus	^V
Löschen links	DEL
Löschen Zeichen	^G
Löschen Wort rechts	^T
Zeile einfügen	^N
Zeile löschen	^Y
Zeilenrest löschen	^Q^Y löscht ab Cursor bis Zeilenende

A.3 Block-Kommandos

Blockbeginn	^K^B
Blockende	^K^K
Marken löschen	^K^H
Wort markieren	^K^T
Block kopieren	^K^C Kopieren an Cursorposition
Block verschieben	^K^V Verschieben an die Cursorposition
Block löschen	^K^Y
Block lesen	^K^R Text von einem angeforderten File lesen
Block schreiben	^K^W Block in ein File schreiben

A.4 Verschiedene Kommandos

Edieren beenden	^K^D Rückkehr zum TURBO-Grundmenü
Indent Ein/Aus	^Q^I automatisches Einrücken der Zeilenanfänge
Restore Zeile	^Q^L Alle Veränderungen in der Zeile werden rückgängig gemacht.
Suchen <Find>	^Q^F Es wird zu suchendes Muster angefordert; Optionen: B rückwärts suchen; G globales Suchen; n Suchen des n-ten Auftretens; U Ignorieren Groß-/Kleinbuchstaben; W nur Worte suchen
Substituieren	^Q^A Zu suchendes Muster und Text werden angefordert; Optionen: B rückwärts suchen; G globales Suchen (top to down); n Suchen des n-ten Auftretens; N Ersetzen ohne Fragen: (Replace (Y/N) ?); U Ignorieren Groß-/Kleinbuchstaben; W nur Worte suchen; Optionen ohne Zwischenraum schreiben und mit <CR> beenden.
Wiederholen	^L Wiederholen des letzten ^Q^F- oder ^Q^A-Kommandos
Abort-Kommando	^U sofortiges Abbrechen jedes Editorkommandos

B. Compiler-Options

○A–,○A+	Retten und Rückspeichern der lokalen Variablen bei Prozeduraufruf ein- bzw. ausschalten, d. h., rekursive Prozeduraufrufe sind bzw. sind nicht möglich.
○C+,○C–	Bei Tastatureingabe werden folgende CTRL-Zeichen interpretiert bzw. ignoriert: ^C Programmabbruch; ^S Stop/Start der Bildschirmausgabe
○I+,○I–	Ein-/Ausschalten der Reaktion des Laufzeitsystems auf E/A-Fehler
○R–,○R–	Überprüfung der Einhaltung des Wertebereiches der Aufzählungs- und Teilbereichstypen ein- bzw. ausschalten (Solche Typen sind Arrayindizes.)
○U+,○U–	Der Abbruch eines Programms durch Betätigen der CTRL-C-Taste wird ermöglicht bzw. verhindert.
○V+,○V–	Bei Parametern vom Datentyp STRING muß die max. Länge des aktuellen mit der des formalen Parameters übereinstimmen bzw. darf unterschiedlich sein.
○Wn	Festlegen der maximalen Verschachtelungstiefe von With-Anweisungen ($0 \leqq n \leqq 9$). In jeder Prozedur werden für With-Anweisungen 2*n Byte im Datenbereich reserviert.
○X+,○X–	Der Zugriff auf Arraykomponenten wird bzw. wird nicht optimiert.

Für alle Options gibt es einen Standard: ○A+, ○C+, ○I+, ○R–, ○U–, ○V+, ○W4, ○X–.

C. Zusätzliche Standardprozeduren und -funktionen

C.1 Ein- und Ausgabe

BLOCKREAD(f,buff,anz)	Lesen von anz Blöcken zu je 128 Byte der Variablen buff von File f. f muß ein typenloses File sein: VAR f: FILE;
BLOCKWRITE(f,buff,anz)	analog BLOCKREAD
CHAIN(f)	Verdrängen des rufenden PASCAL-Programms durch das auf dem Kettenfile f abgespeicherte PASCAL-Programm. f muß ein typenloses File und vorher mit ASSIGN zugewiesen worden sein.
EXECUTE(f)	Wie CHAIN(f), jedoch darf f ein beliebiges COM-File sein.
ERASE(f)	Löschen des f zugewiesenen Files
FLUSH(f)	Leeren des dem File zugeordneten Puffers.
RENAME(f,string)	Das f zugewiesene File erhält den neuen in string angegebenen Namen.

C.2 Bildschirmausgabe und Tastatureingabe

CLREOL	Löschen einer Bildschirmzeile ab Cursorposition
DELLINE	Löschen der gesamten Zeile, in der der Cursor steht
INIT	Ausgabe der Terminalinitialisierungsfolge
INSLINE	Einfügen einer Leerzeile an Cursorposition
KEYPRESSED	Liefert TRUE, falls ein Zeichen im Eingabepuffer der Tastatur

Schluß

Videosteuerung VIS3 mit GDC U 82720 D (Teil 2)

Dr. Wilfried Quednow, Heidrun Bade, Walter Hermann
Akademie der Wissenschaften der DDR,
Zentrum für wissenschaftlichen Gerätebau Berlin

Dieser Teil beschreibt die Nutzung der Videosteuerung VIS3 auf unterschiedlichen Softwareebenen – angefangen von der VIS3-Geräteschnittstelle über die Unterprogrammschnittstelle der Farbgrafikroutinen FGR 186 einschließlich ihrer Anpassung an BASIC und TURBO-PASCAL bis hin zur GSX-Schnittstelle des Gerätetreiberprogramms VGLIB 186. Die Schnittstellen werden überblicksmäßig dargestellt und der jeweils realisierte Leistungsumfang angegeben. (Teil 1 erschien in MP 3/1988.)

Geräteschnittstelle (I/O-Ports)

Die Nutzung dieser Schnittstelle setzt detaillierte Kenntnisse über die Programmierung des Grafik-Display-Controllers GDC U 82720 D voraus und ist im allgemeinen sehr aufwendig. Es sollte nur darauf zurückgegriffen werden, wenn entsprechende Erfahrungen in der hardwarenahen Programmierung vorliegen und die weiter unten beschriebenen Programme FGR 186 bzw. VGLIB 186 nicht ausreichen oder zeitoptimiert werden müssen. In allen übrigen Fällen sollten die Anwenderprogramme eine direkte Bedienung der Geräteschnittstelle vermeiden.

Die Videosteuerung VIS3 belegt 16 aufeinanderfolgende Geräteadressen (Ports), die jedoch nicht alle benutzt werden. Mit den Adressen M0H ... M6H (M = Moduladresse, auf der Steckeinheit über Wickelbrücken einstellbar) können

- der GDC (3 Adressen)
- das Palettenregister PAL-REG
- das Register WRITEN-REG
- das Register ZOOM-REG
- das Steuerregister CTRL-REG

angesprochen werden. Die Adressen M7H ... MFH sind frei. Die Datenübergabe zwischen ZRE des Mikrorechnersystems und VIS3 erfolgt nicht nur über den Datenbus, sondern im Falle des Palettenregisters auch über den höherwertigen Teil des Adreßbusses (AB8 ... AB15). Um unnötige Fehler zu vermeiden, sollten deshalb grundsätzlich nur die Anweisungen IN r,(C) bzw. OUT (C),r verwendet werden, da bei ihnen der höherwertige Teil des Adreßbusses mit dem Inhalt des B-Registers U 880 belegt und somit die zusätzlichen Informationen problemlos übertragen werden können. Die Register sind bei Benutzung dieser Anweisungen nach folgendem Schema zu belegen:

Das r-Register (r = A,D,E,H,L)

D	D	D	D	D	D	D	D
r7	r6	r5	r4	r3	r2	r1	r0

enthält die Daten, die entsprechend dem im C-Register verschlüsselten Kommando K übergeben werden.

Das B-Register

X	X	X	X	R	R	R	R
B7	B6	B5	B4	B3	B2	B1	B0

enthält bei Palettenregisterzugriff die Adresse der Speicherzelle R, die gelesen bzw. beschrieben werden soll.

Das C-Register

M	M	M	M	0	K	K	K
C7	C6	C5	C4	C3	C2	C1	C0

enthält die Moduladresse M und das auszuführende Kommando K.

Weitere Einzelheiten sind der Kommandoübersicht in Tafel 1 zu entnehmen. Es können alle laut Technischem Handbuch /1/ möglichen GDC-Kommandos für die Betriebsart Grafik-Mode einschließlich DMA-Betrieb, benutzt werden.

Tafel 1 Kommandos der Videosteuerung VIS3
Bei fehlenden oder unvollständigen Registerbelegungen haben die betreffenden Bits keine Bedeutung und können beliebig ergänzt werden.

Kommando	Registerbelegung
Kommando in GDC-FIFO schreiben	C7–C0: MMMM0000 r7–r0: GDC-Kommandobyte
GDC-FIFO lesen	C7–C0: MMMM0000 r7–r0: Datenbyte aus GDC-FIFO
Parameter in GDC-FIFO schreiben	C7–C0: MMMM0001 r7–r0: GDC-Parameterbyte
GDC-Status lesen	C7–C0: MMMM0001 r7–r0: GDC-Statusbyte
Maske in Register WRITEN-REG schreiben	C7–C0: MMMM0010 r4: erlaubt (r4=1) bzw. sperrt (r4=0) das BWS-Schreibsignal für Bildebene 0 r5: dto. für Bildebene 1 r6: dto. für Bildebene 2 r7: dto. für Bildebene 3
Farbe in Palettenregister PAL-REG schreiben	C7-C0: MMMM0011 B3–B0: Position in der Farbtabelle (B3=LSB) r4: schaltet „Rot" ein (r4=0) bzw. aus (r4=1) r5: schaltet „Grün" ein (r5=1) bzw. aus (r5=0) r6: schaltet „Blau" ein (r6=0) bzw. aus (r6=1) r7: schaltet volle Helligkeit (r7=0) bzw. halbe Helligkeit (r7=1) ein
Farbe aus Palettenregister PAL-REG lesen	C7–C0: MMMM0011 B3–B0: Position in der Farbtabelle (B3=LSB) r7–r4: Farbe (gleiche Codierung wie beim Schreiben)
Steuerbyte in Steuerregister CTRL-REG schreiben	C7–C0: MMMM0100 r4: schaltet DACK-Eingang des GDC auf Hoch-Pegel (r4=0) bzw. Tief-Pegel (r4=1) r5: erlaubt den Zugriff zum Palettenregister ständig (r5=0) bzw. nur während des Zeilenrücklaufs r6: VSYNC intern (r6=0) bzw. VSYNC extern (r6=1) r7: schaltet Monitorsignale ein (r7=1) bzw. aus (r7=0)
Zoomfaktor in das Zoomregister ZOOM-REG schreiben	C7–C0: MMMM0101 r7–r4: Zoomfaktor (r4=LSB)
DRQ-Signal lesen	C7–C0: MMMM0110 r0: DRQ liegt auf Hoch-Pegel (r0=0) bzw. Tiefpegel (r0=1)

Farbgrafikroutinen FGR 186

Die Farbgrafikroutinen bilden die erste Ebene in der Hierarchie grafischer Software und sind nichts anderes als eine Unterprogrammbibliothek auf Assemblerniveau (Speicherplatzbedarf etwa 3 KByte). Mit Hilfe eines Linkers kann sie in jedes in Assemblersprache (Z-80-Mnemonik) geschriebene Anwenderprogramm eingefügt werden. Im wesentlichen werden von diesen Routinen folgende Grundfunktionen realisiert:

- Ausführung von Initialisierungs- und Steuerungsaufgaben, wie Anfangsinitialisierung des GDC, Bildverschiebung usw.
- Zeichnen elementarer grafischer Objekte auf dem Bildschirm, wie Vektoren, Kreise usw.

Das Kernpropblem besteht dabei darin, die gewünschte Funktion in Steuersequenzen umzusetzen und diese anschließend an den GDC zu übergeben. Des weiteren wird eine Koordinatentransformation vorgenommen, die die vom GDC verlangte eindimensionale Adressierung des Bildwiederholspeichers in eine anwenderfreundliche zweidimensionale Adressierung der Form X,Y umwandelt. Der Koordinatenursprung liegt in der linken unteren Bildschirmecke.

Schnittsstelle

Die Schnittstelle der Farbgrafikroutinen ist mit Rücksicht auf kurze und universelle Anwendbarkeit bewußt einfach gestaltet und nicht an einem grafischen Standard orientiert.

Der Aufruf der einzelnen Unterprogramme erfolgt über die Startadresse der gewünschten Routine im Sprungverteiler. Ein direkter Aufruf der Routinen unter Umgehung des Sprungverteilers sollte aus Kompatibilitätsgründen vermieden werden. Zur Parameterübergabe werden entweder Register oder Arbeitszellen benutzt. Für alle Routinen mit Ausnahme von PZ (siehe Tafel 2) gilt, daß nach Abarbeitung nur der Flag- und A-Registerinhalt verändert sind, während alle übrigen Registerinhalte erhalten bleiben.

Leistungsumfang

Die von den Farbgrafikroutinen realisierten Softwarefunktionen sind so angelegt, daß die Hardwaregegebenheiten der VIS3 möglichst voll ausgenutzt werden. Das heißt, die Freiheitsgrade der Hardware, wie unterschiedliche Bildwiederholspeichergrößen, und der Anschluß verschiedener S/W- und Farbmonitore bzw. Fernsehgeräte werden voll berücksichtigt und alle GDC-Kommandos der Betriebsart Grafik-Mode unterstützt.

Die Einstellung der gewünschten Bildwiederholspeichergröße und des verwendeten Monitors/Fernsehgerätes erfolgt in der Initialisierungsphase, wobei die Standardkonfiguration mit einem Bildwiederholspeicher von 512∗512 Punkten und dem Farbmonitor MON K 7226 voreingestellt ist. Änderungen dieser Werte können dann mit dem zum FGR 186 zugehörigen Installationsprogramm INSTFGR vorgenommen werden. Die Größe des auf dem Monitor sichtbaren Bildfeldes wird von der Bild- und Zeilenfrequenz des Monitors und der Taktfrequenz der VIS3 bestimmt. Sie ist damit konfigurationsabhängig und beträgt bei der Standardkonfiguration z. B. 384(x) ∗ 288(y) Punkte. Einen Überblick über die vom Anwender nutzbaren Routinen gibt der Sprungverteiler in Tafel 2.

Tafel 2 Sprungverteiler der Farbgrafikroutinen FGR 186

Aufruf	Adresse	Funktion	Parameter	UP
AIN	STADR+00	Initialisierung	Parameterfeld ab dmode	DI,AR3
ABL	STADR+03	Bild in Farbebene löschen	Reg.A – Farbmaske	DU,AR3
ACU	STADR+06	Cursor setzen	Reg.DE- X-Koordinate Reg.HL- Y-Koordinate	DI,AR3
AVA	STADR+09	Vektor ausgeben	GPX1,GPY1 – Punkt 1 GPX2,GPY2 – Punkt 2	DI,AR3 LG
AZA	STADR+12	Textzeichen ausgeben	Reg.C – Zeichen im ASCII-Code	DI,AR3 TG,ZL3
AZO	STADR+15	ZOOM-Faktoren einstellen	GZOOM – Grafikzoom TZOOM – Textzoom	DI,AR3
APA	STADR+18	Punkt ausgeben	GFARB – Farbmaske Reg.DE- X-Koordinate REG.HL- Y-Koordinate	DI,AR3 PU
AKR	STADR+21	Kreis ausgeben	KRMX,KRMY – Mittelpunkt KRAD – Radius KRS – Radius*sin(π/4)	DI,AR3 KR
ALI	STADR+24	Linie ausgeben (waagerecht, senkrecht)	LX1,LY1 – Punkt 1 LX2,LY2 – Punkt 2	DI,AR3 LI
AVE	STADR+27	Vektor mit Farbe und Muster ausgeben	Reg.A – Farbmaske Reg.BC – Muster GPX1,GPY1 – Punkt 1 GPX2,GPY2 – Punkt 2	DI,AR3 LG
ASM	STADR+30	Ausgabemode setzen	SM – Ausgabemode	DI,AR3
APZ	STADR+33	Polygonzug ausgeben	PANZ – Punktanzahl PPUF – Pufferadresse	DI,AR3 PZ,LG
AAC	STADR+36	Alphacursor positionieren	TZEI – Textzeile TSPA – Textspalte	DI,AR3 AC
ATA	STADR+39	Textstring ausgeben	TANZ – Zeichenanzahl TPUF – Pufferadresse	DI,AR3 TA,TG
AUF	STADR+42	Farbmaskenregister setzen	Reg.A – Farbmaske	DI,AR3
AUP	STADR+45	Linienmuster einstellen	Reg.BC – Linienmuster	DI,AR3
ATI	STADR+48	Textrichtung setzen	TDIR – Textrichtung	DI,AR3
AWI	STADR+51	Bildfenster-positionierung ausgeben	Reg.A – Fensteranzahl BFX1,BFY1 – Fenster 1 BFX2,BFY2 – Fenster 2	DI,AR3 WI2
ALO	STADR+54	Farbtabelle ändern und ausgeben	Reg.A – Farbe Reg.HL – Tabellenposition	DI,AR3
APL	STADR+57	Punkt lesen (Farbcode in Register A und GFARB)	Reg.DE- X-Koordinate Reg.HL- Y-Koordinate	DI,AR3 PL
ARS	STADR+60	Rechteck ausgeben	GDIR – Richtung RAS – Länge in Richtung RSB – Breite	DI,AR3 RS
ARF	STADR+63	Rechteck mit Muster füllen	Reg.BC – Musteradresse GDIR – Richtung RSA – Länge in Richtung RSB – Breite	DI,AR3 RF
ADR	STADR+66	mit DMA in Puffer lesen	DZEI – Zeilenzahl DSPA – Spaltenzahl DPUF – Pufferadresse	DI,AR3 DMA
ADW	STADR+69	mit DMA aus Puffer ausgeben	DZEI – Zeilenzahl DSPA – Spaltenzahl DPUF – Pufferadresse	DI,AR3 DMA
ALS	STADR+72	Lichtstiftposition lesen	Reg.A – Farbe Reg.DE – X-Koordinate Reg.HL – Y-Koordinate	DI1,AR2 LS

Tafel 3 Standardfunktionen des GSX-80

Opcode	Definition
01	Open Workstation
02	Close Workstation
03	Clear Workstation
04	Update Workstation
05	Escape
	Id Definition
	01 Inquire addressable character cells
	02 Enter graphics mode
	03 Exit graphics mode
	04 Cursor up
	05 Cursor down
	06 Cursor right
	07 Cursor left
	08 Home Cursor
	09 Erase to end of screen
	10 Erase to end of line
	11 Direct cursor address
	12 Output cursor addressable text
	13 Reverse video on
	14 Reverse video off
	15 Inquire current cursor address
	16 Inquire tablet status
	17 Hardcopy
	18 Place cursor at location
	19 Remove cursor
06	Polyline
07	Polymarker
08	Text
09	Filled area
10	Cell array
11	Generalized drawing primitive (BAR, ARC, PIE SLICE, CIRCLE)
12	Set character height
13	Set character up vector
14	Set color representation
15	Set polyline linetype
16	Set polyline width
17	Set polyline color index
18	Set polymarker type
19	Set polymarker scale
20	Set polymarker color index
21	Set text font
22	Set text color index
23	Set fill interior style
24	Set fill style index
25	Set fill color index
26	Inquire color representation
27	Inquire cell array
28	Input locator
29	Input valuator
30	Input choice
31	Input string
32	Set writing mode
33	Set input mode

Tafel 4 Zusatzfunktionen des Gerätetreiberprogramms VGLIB 186

Opcode	Definition
05	Escape
	Id Definition
	20 Hardware zoom
	21 Define hardware zoom window
	22 Display on
	23 Display off
	24 Pan up
	25 Pan down
	26 Pan right
	27 Pan left
	28 Pan home
	29 DMA read
	30 DMA write
	31 Virtual addressing on
	32 Device addressing on
	33 Absolute addressing on
	34 Relative addressing on
	35 Define relative addressing point
	36 Define circle eccenticity
	37 Double-Y on
	38 Double-Y off
	39 Define portaddress

Tafel 5 Technische Daten der VIS3 (siehe auch MP 3/88, Seite 66, Tafel 1)

Displayart:	vollgrafisches farbtüchtiges Rasterdisplay mit GDC U 82720 DC04 (Grafikmode)
Farben:	16 (erweiterbar durch Kaskadierung mehrerer VIS3)
Auflösung des Monitorbildfeldes:	384 (hor.) * 288 (vert.) Punkte[1]
Adressenbelegung:	16 Geräteadressen (über Wickelbrücken einstellbar)
Bildwiederholspeicher (BWS):	
– Größe	je nach Bestückung: 256 * 256 * 4 Bit (mit K 565 RU6) 512 * 512 * 4 Bit (mit U 2164 C25) 1024 * 1024 * 4 Bit (mit MK 41256)
– Zugriff	über GDC (alle GDC-Kommandos einschließlich DMA nutzbar)
– Zugriffsverfahren	Cycle-Stealing-Prinzip (Sperrung des Zugriffs im sichtbaren Teil der Displayzeilen möglich)
– Displayzyklus	~ 400 ns
– Lese-/Schreibzyklus (RMW-Zyklus)	~ 800 ns
– Schreibmodi	replace, set, reset, complement
– Grafische Operationen	Linie, Rechteck, Kreisbogen, Text, Flächen füllen

BASIC- und TURBO-PASCAL-Anpassung

Anwenderprogramme für die Videosteuerung VIS3, die in BASIC oder TURBO-PASCAL geschrieben werden sollen, können ebenfalls die Farbgrafikroutinen benutzen. Voraussetzung dafür ist, daß mit einem CP/M-kompatiblen Betriebssystem gearbeitet wird und die zum FGR 186 gehörenden Interfaceprogramme BIVIS3 (für BASIC) bzw. TUVIS3 (für TURBO-PASCAL) zusätzlich geladen werden. Die Interfaceprogramme sind dem FGR 186 vorgeschaltet und erledigen den Aufruf der Farbgrafikroutinen sowie die Parametervermittlung zwischen der jeweiligen Hochsprache und dem FGR 186. Der Aufruf in BASIC erfolgt mit

CALL NAME (PARAMETER 1, PARAMETER 2...)

und in TURBO-PASCAL mit

NAME (PARAMETER 1, PARAMETER 2...);

NAME ist dabei eine Variable mit der Startadresse der gewünschten Routine im BIVIS3 bzw. eine externe Prozedur im TUVIS3 und im Anwenderprogramm gesondert zu vereinbaren. Dazu steht eine entsprechende Liste bzw. Include-Datei zur Verfügung.

Bei der praktischen Nutzung unter BASIC ist zu beachten, daß zuerst das BIVIS3 und anschließend BASIC mit eingeschränktem Speicherbereich geladen werden müssen. Das ist erforderlich, da BIVIS3 an das Ende des Arbeitsspeichers (TPA) ab Adresse 0B000H geladen wird und sonst von BASIC überschrieben werden würde. Bei TURBO-PASCAL schließt sich das TUVIS3 unmittelbar an die Laufzeitbibliothek an. Anwenderprogramme sind deshalb als CHN-Files mit Startadresse 3000H zu übersetzen und mit PAS3XXX PROGRAMMNAME aufzurufen.

Gerätetreiberprogramm VGLIB 186

Die nächste, über den grafischen Routinen liegende Schicht ist die Nutzeroberfläche, die im allgemeinen von Gerätetreiberprogrammen realisiert wird. Im Gegensatz zu den Ausführungen des vorhergehenden Abschnitts wird sie weniger von den Gegebenheiten der Hardware beeinflußt, als vielmehr von den funktionellen Forderungen der Anwenderprogramme bzw. des Betriebssystems, unter denen diese Anwenderprogramme laufen. Hier spielen also insbesondere die Fragen grafischer Standards und damit auch die Portabilität grafischer Software – eine „uralte“, aber dennoch in der Praxis schwer zu realisierende Forderung – die entscheidende Rolle.

Das nachfolgend beschriebene Gerätetreiberprogramm VGLIB 186 (VIS3 Graphic Library; Speicherplatzbedarf etwa 14 KByte) orientiert sich an dem *Graphic Kernel System* (GKS) /2/, /3/ und realisiert eine Schnittstelle entsprechend dem GSX-80 /4/, /5/, /6/, das von der Firma Digital Research als grafische Erweiterung ihres Betriebssystems CP/M erarbeitet und vertrieben wurde. Der CP/M-Philosophie folgend, den hardwareunabhängigen Teil (BDOS) und den hardwareabhängigen Teil (BIOS) voneinander zu trennen, wird auch hierbei eine Einteilung in das sogenannte

GDOS – hardwareunabhängiger Teil (Koordinatentransformation NDC → DC)
GIOS – hardwareabhängiger Teil (Ausführung grafischer Befehle und Steuerkommandos)

vorgenommen.

Durch VGLIB 186 wird sowohl das GIOS als auch das GDOS realisiert. Es kann deshalb nicht nur unter GSX-80 laufen, sondern auch völlig unabhängig vom Betriebssystem direkt mit dem Anwenderprogramm verbunden werden. Welche Möglichkeiten der Anwender wählt, ist generierungsabhängig und mit Hilfe des VGLIBIN-Files, das beim Binden in jedem Fall mit einbezogen werden muß, einstellbar. Natürlich können damit ähnlich dem INSTFRG auch weitere Einstellungen bezüglich der Initialisierung des GDC, der Geräteadresse u. ä. vorgenommen werden.

Schnittstelle

Der direkte Aufruf von VGLIB 186 (ohne GSX-80) ist mit einem Unterprogrammsprung der Form

CALL VGLIB##

vorzunehmen, wobei der Name im Anwenderprogramm als externe Marke zu kennzeichnen ist. Im GSX-80 erfolgt der Aufruf über die BDOS-Funktion 115.

Für die Parametervermittlung wird ein Parameterblock benutzt, der die Adressen fünf verschiedener Ein-/Ausgabefelder enthält:

PB Adresse des Control-Feldes ‚contrl‘
PB+2 Adresse des Eingangsparameterfeldes ‚intin‘
PB+4 Adresse des Eingangskoordinatenfeldes ‚ptsin‘
PB+6 Adresse des Ausgangsparameterfeldes ‚intout‘
PB+8 Adresse des Ausgangskoordinatenfeldes ‚ptsout‘

Alle Daten, die über diese Felder übergeben werden, sind 16-Bit-Integerzahlen (Ausnahme: DMA read, DMA write). Die Adresse des Parameterblockes ist bei Aufruf im Registerpaar DE einzutragen.

Leistungsumfang

Das Gerätetreiberprogramm VGLIB 186 realisiert alle Funktionen, die in der Grafiknorm des GSX-80 festgelegt sind. Einen Überblick über die Standardfunktionen gibt Tafel 3. Neben diesen Standardfunktionen erlaubt GSX-80 auch die Erweiterung des Funktionssatzes durch nutzer- bzw. hardwarespezifische Befehle mit Hilfe weiterer ESCAPE-Codes. Um die Möglichkeiten des GDC voll auszunutzen, wurde auch davon Gebrauch gemacht und der Funktionssatz um die in Tafel 4 aufgeführten Zusatzfunktionen ergänzt.

Einsatzerfahrungen

Praktische Erfahrungen beim Einsatz der Videosteuerung VIS3 und deren Softwarekomponenten wurden im ZWG Berlin, im ZEG Mittweida und an der HfV Dresden auf dem Gebiet der Meßtechnik, d. h. der Darstellung von Meßkurven gesammelt. Dabei zeigte sich, daß Bildauflösung und Geschwindigkeit des Bildaufbaus in den meisten Fällen als gut eingeschätzt wurden. Hinsichtlich der benutzten Software wurde eindeutig TURBO-PASCAL in Verbindung mit den Farbgrafikroutinen FGR 186 favorisiert.

Die Anwendung des Gerätetreiberprogramms VGLIB 186 für diese Zwecke brachte zwar einerseits die bereits erwähnte Softwareportabilität – hier konkret zwischen dem GSX-kompatiblen Betriebssystem SCPX des A 7100 und dem 8-Bit-Mikrorechner (PC 1715, BC A 5120 usw.) – andererseits aber den Nachteil des großen Speicherplatzbedarfs bzw. zu langer Programmlaufzeiten. Zwischen diesen beiden Kriterien liegt letztlich auch der Entscheidungsspielraum des potentiellen Anwenders, dieses System im konkreten Fall einsetzen zu können.

Literatur

/1/ GDC U 82720 – Technische Beschreibung VEB Mikroelektronik „Karl Marx“ Erfurt, 1987
/2/ Harbeck, G.: GKS – die Grafiknorm Elektronik 33 (1984) 21, S. 61–64 (Teil 1) Elektronik 33 (1984) 22, S. 123–127 (Teil 2) Elektronik 33 (1984) 23, S. 143–146 (Teil 3)
/3/ Enderle, G.; Kansy, K; Pfaff, G.: Computer Graphics Programming, Springer-Verlag Berlin, 1983
/4/ GSX-80, Graphics Extension Programmer's Guide Digital Research, USA
/5/ Schlöter, M.: GSX – die unbekannte Größe Pascal 1 (1987) 1, S. 52–64 (Teil 1) Pascal 2 (1987) 2, S. 82–85 (Teil 2)
/6/ Scott, J.: Graphics device drivers – Drivers free users from device dependence Mini-Micro Systems XIX (1986), S. 19–27

✉ KONTAKT ☏

Akademie der Wissenschaften der DDR, Zentrum für wissenschaftlichen Gerätebau, Rudower Chaussee 6, Berlin, 1199; Tel. 6743381

Schluß

Übersetzungstechniken für Mikroprozeßrechnersprachen

Dr. Mirko Zanter, Prof. Dr. Dr. Michael Roth
Technische Hochschule Ilmenau,
Sektion Technische und Biomedizinische Kybernetik

Eigenschaften einer Mikroprozeßrechnersprache

Der umfassende Einsatz von Mikroprozeßrechnern in der Labor- und Prozeßautomatisierung erfordert die Entwicklung und Implementierung hochleistungsfähiger Übersetzer von Mikroprozeßrechnersprachen (zum Beispiel PEARL /1/, Concurrent-PASCAL /2/, PLZRTC /3/, MP-BASIC /4/). Solche Sprachen und ihre Laufzeitsysteme sind durch folgende Eigenschaften gekennzeichnet /5/, /6/:

- Ausdrucksmittel zur Koordinierung paralleler Rechenprozesse
- Effektive Beschreibungsmittel für numerische und nichtnumerische Algorithmen
- Strukturierte Programmsteueranweisungen und modularer Programmaufbau
- Unterbrechbare und wiedereintrittsfähige Laufzeitprogramme, die mit einer dem Anwendungszweck entsprechenden (relativ) hohen Geschwindigkeit abgearbeitet werden können
- Hohe Portabilität der Laufzeitprogramme durch Implementierung in eine standardisierte Softwareumgebung
- Einbindung maschinennaher Systemprogramme über anwendergerechte Schnittstellen auf symbolischem Niveau für eine einfache Arbeit mit der Prozeßperipherie
- Übereinstimmung der Sprachkonstruktionen mit grafischen Beschreibungsmitteln (insbesondere Struktogramme und Petri-Netze) zur Unterstützung des Programmentwurfs.

Ein breiter Anwenderkreis mit dem unterschiedlichsten Ausbildungsstand auf dem Gebiet der Softwareentwicklung erfordert eine nutzerfreundliche interaktive Programmerstellung.

Turbo-Konzept

Die Eigenschaften einer Mikroprozeßrechnersprache haben einen direkten Einfluß auf die Gestaltung der Übersetzungstechniken. Insbesondere die letzte Forderung nach interaktiver Programmerstellung kann, sofern dies der Aufbau der zu übersetzenden Sprache erlaubt /7/, /4/, mit Hilfe des Turbo-Konzepts erfüllt werden (Bild 1).

Dabei ist der Einsatz eines syntaxgesteuerten Editors zur zeilenweisen (inkrementellen) Eingabe und lexikalischen/syntaktischen Analyse des Quellprogramms vorteilhaft, weil auftretende Syntaxfehler sofort interaktiv erkannt und korrigiert werden können (lokale Syntaxprüfung). Nach beendeter Programmeingabe startet ein Compilerpaß, der u. a. die Adreßzuweisung für alle statischen Datenobjekte und Programmsteueranweisungen vornimmt (globale Syntaxprüfung).

Der auf diese Weise erzeugte Zwischencode ist aufgrund der stark verminderten Organisationszeiten mit einer hohen Geschwindigkeit interpretierbar. Erkennt dabei der Zwischencodeinterpreter einen Laufzeitfehler im Programm, wird unter Einschaltung eines

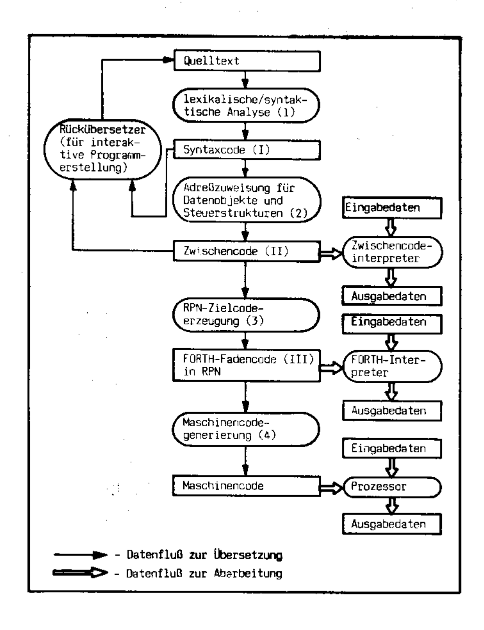


Bild 1 Datenflußgraph eines mehrstufigen Compiler-/Interpretersystems
(1) bis (4) – Compilerpässe
(I) bis (III) – Zwischenprogramme
RPN – reverse (umgekehrte) polnische Notation

Rückübersetzers die fehlerhafte Stelle im Programm angezeigt, so daß ebenfalls mit Hilfe des syntaxgesteuerten Editors eine sofortige Programmkorrektur möglich ist. Die Rückübersetzung wird durch den spezifischen Aufbau des Zwischencodes gesichert /4/. Das Turbo-Konzept ermöglicht einen schnellen (internen) Wechsel zwischen den Systemprogrammen und trägt damit zu einer anwendergerechten interaktiven Programmerstellung bei.
Auf die Probleme der lexikalischen/syntaktischen Analyse soll an dieser Stelle nicht näher eingegangen werden /8/, /9/, /10/. An die Syntaxanalyse schließt sich die semantische Synthese (Maschinencodeerzeugung) der zu übersetzenden Programmiersprache an.

FORTH als Systemimplementierungs- und Zwischensprache

Bei der semantischen Synthese ist die Anwendung der reversen (umgekehrten) polnischen Notation (RPN) aufgrund ihres strukturellen Aufbaus zur Formulierung der Programme in einer weiteren Zwischensprache besonders gut geeignet und in der Compilertechnik verbreitet /8/, /11/. Um eine hohe Portabilität bei der Implementierung dieser Zwischenprogramme zu erreichen, ist deren Formulierung in einer standardisierten Zwischensprache nötig, die das Konzept der reversen polnischen Notation konsequent unterstützt. Dazu kann FORTH /12/, /13/, /14/, /15/ vorteilhaft verwendet werden. FORTH ist die Sprache eines virtuellen Prozessors. Darüber hinaus bildet FORTH ein intergriertes Programmiersystem. Die Übersetzung des syntaktisch analysierten Zwischencodes (Bild 1) kann in einen FORTH-Quelltext oder direkt in den FORTH-Fadencode erfolgen.
Insbesondere durch die Möglichkeit der maschinennahen Programmierung bei gleichzeitigem Vorhandensein von Elementen höherer Sprachen, verbunden mit der guten interaktiven Testunterstützung durch das Programmiersystem, ist FORTH zur Systemprogrammierung gut geeignet. Die Verwendung von FORTH als Systemimplementierungs- und Zwischensprache bei der semantischen Synthese einer Mikroprozeßrechnersprache hat folgende Vorteile:

- Infolge des gleichen strukturellen Aufbaus der Systemimplementierungs- und Zwischensprache wird eine effektive Realisierung der Übersetzungstechniken erreicht, was besonders durch die compilierenden Eigenschaften des FORTH-Textinterpreters unterstützt wird.
- Die Abarbeitung der Zwischenprogramme kann mit einem standardisierten FORTH-Adreßinterpreter erfolgen. Aufgrund des spezifischen strukturellen Aufbaus der FORTH-Adreßparameterliste (Fadencode) wird eine hohe Abarbeitungsgeschwindigkeit der kompakten Zwischenprogramme ermöglicht, die für viele Probleme (dem Anwendungsgebiet des zu übersetzenden Programms entsprechend) bereits ausreicht, so daß eine weitere Übersetzung nicht nötig ist.
- Das FORTH-Programmiersystem stellt eine genormte Softwareschnittstelle zur Verfügung, die eine hohe Portabilität der Übersetzer- und Zwischen-(Ziel-)programme gewährleistet.
- Durch die Universalität des FORTH-Konzepts wird die Entwicklung des Laufzeitsystems erleichtert.
- Eine interaktive Arbeitsweise ist sowohl zum Zeitpunkt des Compilerbaus als auch beim Test der Zwischenprogramme möglich.
- Maschinennahe Anwenderprogramme zur Arbeit mit der Prozeßperipherie können mit FORTH effektiv realisiert und über symbolische Schnittstellen als Prozeduren in die Mikroprozeßrechnersprache eingebunden werden.
- Zur weiteren Übersetzung der (FORTH-) Zwischenprogramme in den Maschinencode eines konkreten Prozessors (Bild 1) zwecks Laufzeitoptimierung können standardisierte und bereits vorgefertigte Lösungen (z. B. Native-Code-Compiler /15/) Verwendung finden, so daß der Prozeß des Compilerbaus rationalisiert wird.

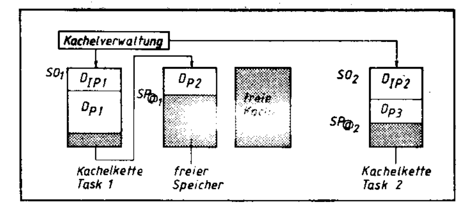


Bild 2 Verwaltung paralleler Taskdatenbereiche mittels Kachelketten
D – Datenstack einer Prozedur

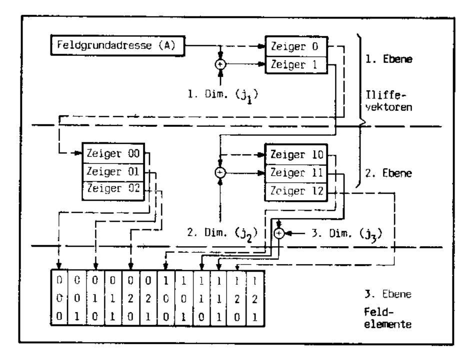


Bild 3 Adressierung von Feldelementen nach dem Iliffe-Verfahren (Beispiel)
Vom Feld $A(i_1, i_2, i_3)$ mit $i_1 = 1$, $i_2 = 2$, $i_3 = 1$ wird das Element $A(j_1, j_2, j_3)$ mit $j_1 = 1$, $j_2 = 1$, $j_3 = 1$ angesprochen.

Tabellengesteuerte Übersetzungsverfahren

Durch die Anwendung tabellengesteuerter Übersetzungsverfahren kann eine weitgehende Abkopplung der Sprachbeschreibung von den Algorithmen des Compilers auf jeder Ebene des Übersetzungsprozesses erreicht werden. Bei einer Sprachkorrektur bzw. -erweiterung ist dann keine Änderung der Übersetzerprogramme nötig. Die Übersetzungstabellen enthalten eine linearisierte Darstellung der syntaktischen (z. B. linearisierte Syntaxdiagramme /4/) und semantischen Regeln der zu übersetzenden Sprache. Die Semantik ist auf der Grundlage des Verfahrens der operationellen Definition der Semantik /17/ in modular aufgebauten Tabellen formulierbar, wobei die semantischen Interpretationsregeln durch einfache Elementaroperationen der virtuellen FORTH-Maschine dargestellt werden können /18/. Die Zuordnung der semantischen zu den syntaktischen Regeln geschieht dabei aufgrund des strukturellen Aufbaus der Syntaxdiagramme. Dieses Verfahren wird durch den streng modularen Charakter von FORTH-Programmen unterstützt.

Aufgaben des Laufzeitsystems Dynamische Speicherverwaltung

Die gegebene Programmstruktur einer zu übersetzenden Mikroprozeßrechnersprache stellt spezifische Anforderungen an die Datenstruktur der Laufzeitprogramme. Dabei ist die Organisation des lokalen Datenbereichs einer Prozedur in Form eines Stacks sinnvoll, damit zur Laufzeit dynamische Datenobjekte (z. B. dynamische Felder) verwaltet werden können. Der lokale Datenbereich eines Tasks vereinigt die lokalen Datenstacks der ihm zugeordneten Initialprozedur und aller von der Initialprozedur zur Bildung einer modularen Programmstruktur aufgerufenen Prozeduren. Die Forderung nach unterbrechbaren und wiedereintrittsfähigen Programmen bei Multitaskbetrieb und bei rekursivem Aufruf bedingt, daß jede Prozedur bei jedem Aufruf einen neuen lokalen Datenstack eröffnen muß. Die lokalen Daten einer unterbrochenen Prozedur dürfen nicht zerstört werden. Aus diesem Grunde sind alle lokalen Stacks der Prozeduren innerhalb eines Tasks selbst stackartig miteinander verbunden. Die Datenbereiche der Tasks existieren parallel zueinander.
Zur Verwaltung der (parallelen) lokalen Datenbereiche der Tasks mittels Kachelketten /11/ (Bild 2) ist ein relativ geringer Speicheraufwand bei gleichzeitig guten Zugriffseigenschaften auf die lokalen Datenobjekte erforderlich. Bei diesem Verfahren wird die Kette von Speichersegmenten gleicher Größe (Kachelkette) abgebildet. Ein zusätzlicher Speichergewinn wird durch die Vereinigung von physisch im Speicher benachbarten Kacheln und durch Aufteilung des lokalen Datenstacks einer Prozedur auf zwei Kacheln erzielt /18/.

Übersetzung von Datenfeldern

Bei der Übersetzung von Feldern kann durch folgende Verfahren eine Verkürzung der Abarbeitungszeit der Laufzeitprogramme erzielt werden:

- Dimensionierung statischer Felder bereits während der Übersetzung im Zusammen-

hang mit der getrennten Behandlung statisch und dynamisch dimensionierter Felder durch den Compiler

- Feldzugriff nach dem Iliffe-Verfahren /19/. Dabei wird zusätzlich zum Vektor der Feldelemente noch eine Zeigerstruktur abgespeichert (Bild 3), deren Aufbau von Anzahl und Größe der Feldindizes abhängt. Diesem geringen zusätzlichen Speicherbedarf steht eine hohe Geschwindigkeit beim Feldzugriff gegenüber, weil zur Berechnung der Adresse des Feldelementes im Gegensatz zum Dope-Verfahren /8/ keine Multiplikationen nötig sind.

Implementierung von Echtzeitanweisungen

Zur Unterstützung der Multitaskverarbeitung stehen in einer Mikroprozeßrechnersprache Ausdrucksmittel zur Koordinierung paralleler Rechenprozesse zur Verfügung. Eine äquivalente Beschreibung dieser Sprachelemente als Petri-Netz-Modell ist zur Unterstützung des Programmentwurfs vorteilhaft /20/, /21/. Das Petri-Netz /22/ ist zur Beschreibung paralleler Rechenprozesse, d. h. zum Entwurf, zur Untersuchung der Eigenschaften einer Struktur, zur grafischen Programmierung der Synchronisationsstellen der Rechenprozesse und zur Dokumentation geeignet.
Die Einbeziehung eines standardisierten Echtzeitbetriebssystemkerns in das Laufzeitsystem zur Implementierung der Echtzeitanweisungen ergibt den Vorteil, daß die Portabilität der Laufzeitprogramme erhöht und der Prozeß des Compilerbaus rationalisiert wird. Dabei ist u. a. die Überführung des Taskzustandsmodells und die Anpassung an die Semaphorebehandlungsmechanismen des Echtzeitbetriebssystemkerns erforderlich.

Rationalisierung der Compilerentwicklung

Die vorgestellten Übersetzungstechniken der semantischen Synthese sind am Beispiel der Sprache Mikroprozeßrechner-BASIC /4/ unter Einbeziehung eines fig-FORTH-Systems untersucht worden, wobei ein prinzipieller Funktionsnachweis erbracht wurde /18/. Gegenwärtig werden Arbeiten zur vollständigen Realisierung des Mehrpaßcompilers nach Bild 1 durchgeführt. Dabei ist zu beachten, daß die Entwicklung und Implementierung eines effektiven Compilers außerordentlich komplex und mit einem hohen Arbeitsaufwand verbunden ist. Aus diesem Grunde ist eine durch das vorgestellte Systemkonzept unterstützte mehrstufige Implementierung mit getrennter Interpretierbarkeit der einzelnen Zwischensprachen anzustreben. Um den Prozeß des Compilerbaus zu rationalisieren, sind standardisierte und vorgefertigte Lösungen zur Realisierung der Systemkomponenten zu verwenden. Das wird durch Nutzung der FORTH-Schnittstelle unterstützt (FORTH-Programmiersystem mit Editor, Interpreter, Debugger, Native-Code-Compiler u. a.).
Weiter ist der umfassende Einsatz rechnergestützer Verfahren, z. B. der UNIX-Softwarewerkzeuge LEX /23/ und YACC /24/, für den Compilerbau erforderlich. Dazu gehört künftig auch die Anwendung von Expertensystemen mit nutzerfreundlichen Interfaces.

Literatur

/1/ Kappatsch, A. u. a.: PEARL – Systematische Darstellung für den Anwender. München, Wien: R. Oldenburg Verlag 1979
/2/ Hansen, P. B.: The programming language Concurrent Pascal. IEEE Trans. software engeneering. SE-1. (1975) 6, S. 199
/3/ Antonov, A.: Einsatz der höheren Echtzeitsprache PLZRTC zur Erstellung moderner μR-Steuer-Software. Nachrichtentechnik, Elektronik 33 (1983) 2, S. 55
/4/ Schorrig, H.: Mikroprozeßrechner-BASIC. Dissertation A. Ilmenau: TH Ilmenau, Sektion TBK, WB CT 1986
/5/ Werner, D.: Programmierung von Mikrorechnern: Programmsysteme – Parallele Prozesse – Echtzeitbetriebssysteme. Berlin: VEB Verlag Technik 1983
/6/ Färber, G.: Prozeßrechnertechnik. Berlin (West), Heidelberg, New York: Springer Verlag 1979
/7/ Zwoch, M.: Der Entwurf von inkrementellen Übersetzern für interaktive Programmiersysteme. Dissertation A. Dresden: TU Dresden, Fakultät für Elektrotechnik/Elektronik 1986
/8/ Loeper, H.; Bachmann, P.: Theorie und Technik der Übersetzerprogramme höherer Programmiersprachen. Leipzig: BSB B. G. Teubner Verlagsgesellschaft 1980
/9/ Bauer, F. L.; Eickel, J.: Compiler construction, an advanced course. Lecture notes in computer science. Vol. 21. Berlin (West): Springer Verlag 1976
/10/ Bachmann, P.: Grundlagen der Compilertechnik. Leipzig: BSB B. G. Teubner Verlagsgesellschaft 1975
/11/ Jähnichen, S.; Oeters, Ch.; Willis, B.: Übersetzerbau. Braunschweig: Vieweg-Verlagsgesellschaft 1978
/12/ Krapp, M.; Richter, J.; Schwartz, J.: Eine FORTH-Systemfamilie. Mikroprozessortechnik 2 (1988) 2, S. 53
/13/ Vack, G.-U.: FORTH: Eine außergewöhnliche Softwarekonzeption, Mikroprozessortechnik 1 (1987) 6, S. 163
/14/ Varga, G.; Krapp, M.: FORTH – Erweiterungen um Multitasking und Programmentwicklungskomponenten. 30. Internationales Wissenschaftliches Kolloquium, Vortragsreihe D, Heft 3. Ilmenau: TH Ilmenau, 21. bis 25. 10. 1985, S. 235
/15/ Zech, R.: Die Programmiersprache FORTH. München: Franzig-Verlag 1984
/16/ Büvel, R.: A FORTH Native Code cross compiler for the MC 68000. Dr. Dobb's Journal. 9 (1984) 9, S. 68
/17/ Riedewald, G.; Maluszinski, J.; Dembinski, P.: Formale Beschreibung von Programmiersprachen – Eine Einführung in die Semantik. Berlin: Akademie-Verlag 1983
/18/ Zanter, M.: Entwicklung und Implementierung von Übersetzungstechniken für Mikroprozeßrechnersprachen. Dissertation A. Ilmenau: TH Ilmenau, Sektion TBK, WB CT 1987
/19/ Iliffe, J. K.; Jodeit, J. G.: Dynamic storage allocation. Computer Journal 5 (1962) 2, S. 200
/20/ Fengler, W.: Technik und Entwurf von Mikroprozeßrechnern. Manuskript. Dissertation B. Ilmenau: TH Ilmenau, Sektion TBK, WB CT 1987
/21/ Roth, M.; Fengler, W.: Einchip-Rechner-Schaltkreise. Mikroprozessortechnik 1 (1987) 2, S. 37
/22/ Starke, P. H.: Petri-Netze. Berlin: VEB Deutscher Verlag der Wissenschaften 1980
/23/ Lesk, M. E.; Schmidt, E.: Lex – a lexical analyzer generator. Murray Hill: Bell Laboratories 1975
/24/ Johnson, S.: YACC: yet another compiler-compiler. Murray Hill: Bell Laboratories 1978

⊠ KONTAKT ☏

Technische Hochschule Ilmenau, Sektion TBK, WB CT, Am Ehrenberg, Ilmenau, 6300; Tel. 74509 (Dr. Zanter)

Lehrmittel zu PASCAL

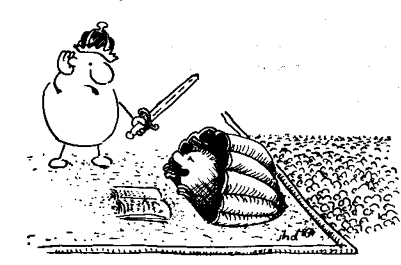
Das Institut für Film, Bild und Ton Berlin gibt seit einigen Jahren Diapositive, Projektionsfolien und Filme für die Informatikausbildung heraus. Vor kurzem erschien ein Lehrmittelkomplex, der das Erlernen des Umgangs mit dem PASCAL-Programmiersystem 88X/X unterstützen kann. Die Projektionsfolien und Diapositive helfen dem Lehrenden, die Effektivität und Anschaulichkeit seiner Lehrveranstaltungen wesentlich zu erhöhen.
Mit der ersten Folienreihe (HFR 945) können die Bestandteile des Programmiersystems PASCAL 880/S, die Bedienoberfläche und die Arbeitsweise beim Programmieren veranschaulicht werden. Grundmenüs, Nebenmenüs und Editierkommandos werden erläutert. PASCAL 88X/X enthält gegenüber TURBO-PASCAL einen leistungsfähigen Systemservice (PLUS-Option), der mit den Folien gezeigt werden kann. Arbeitsabläufe und ein kleines Beispiel sind enthalten.
In den beiden anderen Folienreihen (HFR 946, HFR 947) werden anhand grafischer Darstellungen, Syntaxdiagrammen und Programmbeispielen wichtige Grundzüge des Sprachumfangs von PASCAL 88/X und TURBO-PASCAL bis hin zur Dateiarbeit und dem Umgang mit Pointern erläutert. Die Beispiele und Aufgaben haben allgemeinverständlichen Charakter und können vom Vortragenden entsprechend speziellen Zielstellungen erweitert und ergänzt werden. Auf eine vollständige Abhandlung des Sprachumfangs mit Hilfe von Lehrmitteln wurde verzichtet.
Die drei Folienreihen werden durch Diareihen mit Bildschirmaufnahmen (HR 1568, HR 1569, HR 1570) ergänzt. Sie ermöglichen, einem größerem Kreis von Lernenden auch ohne Computerkabinett die Arbeit mit dem Programmiersystem praxisnah zu demonstrieren. Die Dias gestatten es, die Arbeitsweise an Beispielen zu verdeutlichen, insbesondere das Auffinden von Editier- und Laufzeitfehlern.
Die Lehrmittel sind in begrenztem Umfang über folgenden Vertrieb zu beziehen: DLB Erfurt, BT Audiovisuelle Lehrmittel, Junker-Jörg-Str. 31, Berlin, 1157.
HFR 945 PASCAL Teil 1, 18 Folien, 103,45 M
HFR 946 PASCAL Teil 2, 20 Folien, 171,60 M
HFR 947 PASCAL Teil 3, 24 Folien, 145,20 M
HR 1568–70 PASCAL-Dias, 33 Diapositive, 90,75 M *Fischer*

In eigener Sache

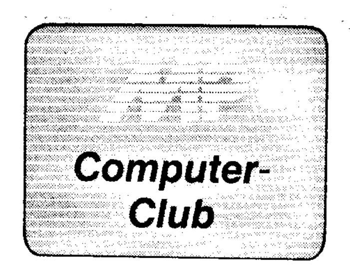
Für eine weitere Redakteurstelle in unserer Zeitschrift suchen wir einen geeigneten Mitarbeiter bzw. eine Mitarbeiterin mit abgeschlossenem Hoch- oder Fachschulstudium und guten Kenntnissen der Computertechnik.
Zu den Aufgaben gehören unter anderem das redaktionelle und fachliche Bearbeiten von Manuskripten, der Besuch und die Auswertung von Fachtagungen, -messen und -ausstellungen, die Zusammenarbeit mit Autoren und Gutachtern sowie ggf. das Testen und Beurteilen von Programmen, die der Redaktion zur Veröffentlichung eingereicht werden.
Falls Sie Interesse an dieser Tätigkeit haben und im Raum Berlin wohnen, rufen Sie uns unter Tel. 2 87 03 71 oder 2 87 02 03 an oder schreiben Sie an:
VEB Verlag Technik, Redaktion MP, Oranienburger Str. 13/14, Berlin, 1020

Kleines Lexikon der Mikrorechentechnik

C wie Coprozessor



Zeichnung: Jens-Helge Dahmen



Verbesserungen des KC 85/4 gegenüber dem KC 85/3

Im VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen wurde der Kleincomputer KC 85/4 entwickelt, welcher im IV. Quartal 1988 produktionswirksam wird und damit den bisher produzierten KC 85/3 ablöst.
Der KC 85/4 stellt eine Weiterentwicklung der Kleincomputerserie aus Mühlhausen dar, dessen verbesserte Eigenschaften im folgenden betrachtet werden.

Vergrößerter Bildwiederholspeicher

Der IRM des KC 85/4 besitzt einen Speicherumfang von 64 KByte, welcher hintereinander beginnend mit Adresse 8000H in vier 16-KByte-Blöcke aufgeteilt ist. Durch Einsatz schneller Speicherschaltkreise sind die Prozessorzugriffe beim Bildaufbau nicht mehr sichtbar.

- „Doppelbild"

Zur Darstellung auf dem Bildschirm enthält der KC 85/4 zwei Bildspeicher (Bild 0, Bild 1), welche wechselseitig beschrieben und angezeigt werden können. Zu jedem Bild gehören ein Farb-, ein Pixel- und ein ASCII-Speicher.

- Byteweise Farbauflösung

Der KC 85/4 hat in diesem Modus eine Farbauflösung von 8*1 Punkten bei 16 Vordergrundfarben und 8 Hintergrundfarben, das heißt, für die Farbinformation von 8 nebeneinander liegenden Bildpunkten ist ein 1 Byte im Farbspeicher reserviert.

- Hochauflösende Grafik (bitweise Farbauflösung)

Bei der Anwendung der hochauflösenden Grafik kann in vier Farben (schwarz, weiß, türkis, rot) jedem Bildpunkt genau eine Farbe zugeordnet werden.

- Lösch- und Überlagerungsmode der Grafikgrundbefehle

Bei der Anwendung der Grafikbefehle bietet der KC 85/4 die Möglichkeit des Löschens einer Linie oder eines Kreises. Weiterhin besteht die Möglichkeit, ein Grafikelement auf ein vorhandenes Bild zu überlagern. Dabei erfolgt nur dort ein Punktsetzen, wo noch keiner existiert; an den Stellen, wo vorher ein Punkt gesetzt war, wird gelöscht.

Vergrößerter RAM

Der Arbeits-RAM des KC 85/4 wurde gegenüber dem KC 85/3 von 16 KByte auf 64 KByte erweitert. Die vier 16-KByte-Speicherblöcke sind in Segmente eingeteilt und können über SWITCH ein- oder ausgeschaltet werden.
Bei der Arbeit in BASIC stehen sofort 48 KByte zur Verfügung. Der letzte 16-KByte-Block kann nur bedingt genutzt werden.

Vergrößerter ROM

Dieser Festwertspeicher umfaßt im KC 85/4 20 KByte. Er beinhaltet das Betriebssystem CAOS 4.1 und den BASIC-Interpreter. Gegenüber dem KC 85/3 wurde der ROM um 4 KByte erweitert. Dadurch war es möglich, die V.24-Grundroutinen ins Betriebssystem einzubinden.

- Betriebssystemerweiterungen

Das Betriebssystem des KC 85/4 wurde um folgende Kommandos erweitert.:

*WINDOW	– Bildschirmfenster einstellen
*MODUL	– Zustand und Struktur der gesteckten Module anzeigen
*SYSTEM	– Anzeige des aktuellen Speicherzustandes
*V24OUT	– Druckertreiberinitialisierung
*V24DUP	– Duplexroutine initialisieren

Die zusätzlich im Betriebssystem enthaltene universelle V.24 Druckerroutine kann alle Drucker der K-6300-Reihe, die Schreibmaschinen der Typenreihe S 6000 und die S 3004 bedienen.
Weiterhin beinhaltet das Betriebssystem eine interruptgesteuerte V.24-Duplexroutine, mit der der KC über eine Tastatur mit V.24-Anschluß oder über einen anderen Computer fernbedient werden kann. Außerdem ist eine Computerkopplung möglich.
Die schnellere Autorepeatfunktion der Tasten ist bei der Menüanwahl über die Cursortasten sowie bei den Editierfunktionen von großem Vorteil. Über eine Speicherzelle kann die Zeitkonstante zwischen Tastenbetätigung und Autorepeatfunktion eingestellt werden.

- Die dritte Tastaturebene

Zum Einbinden weiterer Tastenfunktionen wurde die Umschaltfunktion (ESC) für die 3. Tastaturebene gewählt. Diese ESC-Funktion wurde auf noch freie Tastenfunktion (SHIFT STOP) gelegt. Mit diesem Schritt stehen nun insgesamt 35 mögliche Steuerfunktionen (24 durch Selbsterstellung) zur Verfügung.

Applikationsstelle VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen

Jörg Eichler

RAM
IRM (Pixelebene)
IRM (Farbebene)
BASIC - ROM
CAOS - ROM
Ebene ··· 6 ··· 5 ··· 4 ··· 3 ··· 2 ··· 1
0H 4000H 8000H C000H FFFFH

Bild 1 Speicherübersicht KC 85/4

Menüführung für den PC 1715 in REDABAS

Die Menütechnik ist zu einem wesentlichen Bestandteil der Nutzerführung bei Anwendersoftware geworden. Im folgenden soll eine einfache und ebenso effektvolle Variante vorgestellt werden.
Die meisten Menüs werden über die Eingabe von Ziffern oder Buchstaben genutzt. Auf dem PC 1715 ist aber auch eine andere Möglichkeit gegeben, die Positionierung eines Leuchtbalkens auf der gewünschten Option und Anwahl derselben durch die ENTER-Taste.
Durch diese Methode lassen sich vor allem auch interessante Gestaltungsvarianten auf dem Bildschirm finden. Der Phantasie des Programmierers sind hierbei kaum Grenzen gesetzt. Die Steuerung des Leuchtbalkens erfolgt mittels der Cursortasten, prinzipielle sind aber auch andere Tasten möglich. Ein spezielles Problem tritt hierbei vor allem bei REDABAS auf, da die GET-Anweisung den Inhalt der entsprechenden Variable aus dem Bildwiederholspeicher liest. Gerade die Cursor- oder ENTER-Taste erzeugen aber keinen sichtbaren Code auf dem Bildschirm, und damit würde die Variable immer ein Leerzeichen zugeordnet bekommen.
Dieses Problem wird durch eine kleine Maschinenroutine umgangen, die weiter nichts macht, als die BDOS-Funktion Nr. 1 (Eingabe eines Zeichens von der Tastatur) aufzurufen und den Code der betätigten Taste in einer bestimmten Speicherzelle abzulegen. Von dort kann sie dann durch PEEK (Speichervariable) in REDABAS abgefragt und ausgewertet werden. Der GET-READ-Befehl wird also durch eine CALL-Routine ersetzt.
In Bild 1 sehen Sie ein kleines Beispielprogramm zur Demonstration.

Rene Iffarth

```
************* DEMO-Programm  Leuchtbalkenmenue *************
********* by 'IFFISOFT' Rene Iffarth  Oktober 1987 *********

? chr(131)
** Cursor ausschalten **
set talk off
erase
store chr(27)+chr(94)+chr(64) to norm
store chr(27)+chr(94)+chr(81) to inv
store chr(134) to int
** Bildschirmattribute fuer Invers- und Normaldarstellung **
do while T
   poke 41984,14,01,205,05,0,50,09,164,201,0
** Einschreiben der Maschinenroutine ab Adresse **
** 41984 (freier Bereich bei REDABAS)           **
   set call to 41984
   set colon off
   erase
   @  5,24 say int+"Demonstrationsmenue"
   @  6,25 say "-------------------"+norm
   @ 10,20 say "     Unterprogramm 1     "+norm
   @ 12,20 say "     Unterprogramm 2     "+norm
   @ 14,20 say "     Unterprogramm 3     "+norm
   @ 16,20 say "           Ende          "+norm
   @ 20,18 say int+"Waehlen Sie mit Cursor hoch und tief"
   @ 21,19 say " Quittieren Sie mit der ENTER-Taste"+norm
   store 10 to zeile
   store 1 to hx
 ** Tastaturabfrage und jeweilige Verschiebung des   **
 ** Leuchtbalkens bis die ENTER-Taste gedrueckt wird **
   do while hx<>13
      @ zeile,20 say inv
      @ 22,75
      call
      store peek(41993) to hx
      @ 22,73 say chr(20)
      @ zeile,20 say norm
      do case
         case hx=32 .or. hx=24
              store zeile+2 to zeile
              if zeile>16
                 store 10 to zeile
              endif
         case hx=5
              store zeile-2 to zeile
              if zeile<10
                 store 16 to zeile
              endif
      endcase
   enddo
** Auswertung der Leuchtbalkenposition und entsprechende **
** Verzweigung in  Unterprogramme o.a.                   **
   do case
      case zeile = 10
           do ...
      case zeile = 12
           do ...
      case zeile = 14
           do ...
      case zeile = 16
           clear
           erase
           ? chr(130)
           return
   endcase
enddo
```

Bild 1 DEMO-Programm „Leuchtbalkenmenü"

REASS – eine Ergänzung zum EDAS des KC 85/3

Der Modul M027 DEVELOPMENT der Rechner KC 85/2 bzw. 3 erlaubt eine Rückübersetzung von Maschinenprogrammen nur in Verbindung mit dem Programmteil CDISASS unter Einbeziehung eines Kassettenrecorders. Das Hilfsprogramm REASS erweitert den Programmkomplex um eine befehlsweise bzw. byteweise Rückübersetzung im BOTOM-Modus des EDAS. Zusätzlich wurde eine Routine zur Identifizierung des Prologbytes eingefügt und damit eine Ausgabe des Programmnamens möglich gemacht. Diese Routine wurde auch in einen speziellen Ansprung des Programms DISASS eingefügt.
Das Hilfsprogramm REASS erscheint im EDAS-Menü, wird mit EDASTAST initialisiert (F-Tastenbelegung) und mit REASS adr. (Prologbyte) (Offset) aufgerufen. Der Einsprung erfolgt in einen modifizierten BOTOM-Modus.

Die Anfangsadresse wird als mögliche ORG-Adresse ausgedruckt. Die befehlsweise Rückübersetzung erfolgt mittels der Tasten F3 und F4. Im REASS-Modus sind sämtliche EDIT-Funktionen nutzbar. Die in Zeile 1DH erscheinende aktuelle Adresse erleichtert die spätere Fortsetzung einer unterbrochenen Reassemblierung. Vor Erreichen der Zeile 1DH muß mit Shift Cursor down umgeblättert werden, da sonst die herausrollenden Zeilen nicht übernommen werden. Die beim Umblättern und beim Abbruch eingefügten Adressen erleichtern die Übersicht und lassen sich später im EDIT-Modus mühelos entfernen.
Da die Übersetzung des Programmkopfes in drei Zeilen erfolgt, kann beim Erscheinen des Kopfes in einer der letzten Zeilen gegebenenfalls die oberste Zeile verschwinden. Die Arbeit mit REASS wird mittels der Taste Brk beendet. Als Epilogbyte wird grundsätzlich 01H ausgegeben, 00H aber auch identifiziert. Bei Rückübersetzung des Programms REASS erfolgt eine Fehlinterpretation des Bytes 27H (Hochkomma).
Falls mit Offset, aber ohne Prologbyte, gearbeitet werden soll, ist statt des Prologbytes 0 (Null) einzugeben. Das Hilfsprogramm REASS nutzt Programmteile der Programme EDAS und DISASS und die Unterprogramme:

RBEF Befehl reassemblieren (Taste F3, Code 13H = Stop)
RBYTE Byte reassemblieren (Taste F4, Code 15H)
PROA Prologbyte-, Namen- und Epilogbyteausgabe
ORGADR Ausgabe der Anfangsadresse (ORG-Adr.)
AKTADR Ausgabe der aktuellen Adresse

In die Arbeit mit dem EDAS hat der Autor weitere Programme des DEVELOPMENT (u. a. DISPLAY und DISASS) sowie Ergänzungen (DUMP, FINDBYTES, CLIB, ...) einbezogen. Während CLIB die Namen von Kassettenaufzeichnungen auflistet, erlaubt das Programm FINDBYTES das Auflisten von Bytefolgen und ist damit zum Aufsuchen von Programmanfängen oder zu verändernder Sprungadressen eine wertvolle Hilfe.

Ablauf der Arbeit mit REASS

Modul M027 DEVELOPMENT aktivieren, Programm REASS laden, EDAS aufrufen, REASS mit EDASTAST initialisieren, im TOP-Modus Semikolon und eventuellen Anfangskommentar eingeben, Rückkehr mit Taste Brk ins EDAS-Menü, Aufruf des Hilfsprogramms mit REASS Adresse (Prologbyte (Offset). Die vorgegebene Adresse erscheint als Kommentar auf dem Bildschirm. Mittels der Stop-Taste oder der Taste F3 wird befehlsweise reassembliert. Kurz vor Erreichen der Zeile mit der folgenden aktuellen Adresse ist mit Shift Cursor down umzublättern. Das Rückübersetzen ist mit der Taste Brk zu beenden. Daten lassen sich mittels Taste F4 (Code 15H) als Bytes definieren. Bei druckbaren ASCII-Zeichen erfolgt eine Ausgabe des Zeichens als Kommentar. EDASTAST belegt die Funktionstasten F1 bis F4 mit den Codes 05, 06, 13, 15. Taste F5 schreibt DEFW 0, Taste F6 DEFB 0 (F7 bis FA dienen zur Druckformatierung).
Ehe das Assembler-Listing auf Kassette abgespeichert wird, kann das Listing bearbeitet und im ASM-Modus auf Richtigkeit überprüft werden. Nach Unterbrechung der Arbeit kann die Fortsetzung bei der zuletzt ausgegebenen aktuellen Adresse erfolgen. Editieren ist im REASS-Modus möglich, aber nicht unbedingt zweckmäßig.

Hans-Joachim Zühlsdorff, Arnstadt

Bild 1 Hilfsprogramm REASS

```
;HILFSPROGRAMM R E A S S  VERS.2.1
;FUER KLEINCOMPUTER KC 85/2 BZW /3
;(NUR IN VERBINDUNG MIT DEM MODUL
;M027 DEVELOPMENT NUTZBAR)
;DAS PROGRAMM GESTATTET DAS BEFEHLS-
;WEISE REASSEMBLIEREN VON MASCHINEN-
;PROGRAMMEN IM BOTOM-MODUS INNERHALB
;DES EDAS
;(C) AUGUST 1987 ZUEHLSDORFF
;
;AUFRUF: REASS ADR.(PROLOGBYTE) (OFFSET)
;.......................................
PV1     EQU 0F003H
ARG5    EQU 0B78AH
;
        ORG 0BE00H         ;PROGRAMMADRESSE
;.......................................
;UP ORGADR GIBT ANFANGSADRESSE AUS,DIESE
;KANN ALS ORG-ADR.GENUTZT WERDEN.
;
ORGADR  CALL PV1
        DEFB 23H
        DEFM ' ;ORG 0'
        DEFB 00H
        PUSH IY
        POP HL
        CALL PV1
        DEFB 1AH
        CALL PV1
        DEFB 23H
        DEFB 08H
        DEFM 'H'
        DEFW 0A0DH
        DEFB 00H
        RET
;.......................................
;UP AKTADR GIBT AUF ZEILE 1DH DIE
;FOLGENDE AKTUELLE ADRESSE AUS.
;
AKTADR  PUSH HL            ;AKTUELLE ADR
        PUSH DE
        PUSH BC
        LD HL,(0B7A0H)
        PUSH HL
        LD HL,01D00H
        LD (0B7A0H),HL
        CALL PV1
        DEFB 23H
        DEFM ';'
        DEFB 000H
        PUSH IY
        POP HL
        CALL PV1
        DEFB 1AH
        POP HL
        LD (0B7A0H),HL
        POP BC
        POP DE
        POP HL
        RET
        NOP
;
;HAUPTPROGRAMM REASS VERS.2.1
;BEFEHLSWEISES REASSEMBLIEREN VON
;MASCHINENPROGRAMMEN IM BOTOM-MODUS
;DES EDAS (TASTE F3,F4/ENDE MIT BRK-T.)
;AUFRUF MIT
; REASS ADR. (PROLOGBYTE) (OFFSET)
;(00 ALS PROLOGBYTE  WIRD IGNORIERT)
;
HEAD    DEFW 0D0D0H
        DEFM 'REASS'
        DEFB 01H
RE0     PUSH HL            ;ADR NACH IY
        POP IY
        LD A,(0B781H)
        CP 03H             ;3 ARGUMENTE?
        JR Z,RE00
        LD HL,00000H
        LD (ARG5-4),HL     ;ARG3:=0(K.OFFS)
RE00    CP 01H
        RET C              ;KEIN ARG.
        LD A,0H
        JR Z,RE01          ;OHNE PROLOGBYTE
        LD A,E             ;PROLOGBYTE..A
RE01    LD (ARG5),A
        LD (ARG5+2),A
        AND A
        JR Z,RE02          ;FALLS PROL.00H
        LD A,1H
        LD (ARG5+2),A      ;FLAG=1:PROLOG
RE02    XOR A
        LD (0B7FEH),A      ;HILFSZELLEN
        LD (0B7FFH),A
        CALL 0C58CH        ;BOTOM-MODUS
        CALL 0C19FH        ; EINSTELLEN
        LD A,1FH
        LD (0B79FH),A
RE1     CALL 0C8A1H
        CALL ORGADR        ;AUFRUFADR.
RE2     CALL 0F003H        ;TASTATURABFR
        DEFB 04H
        CP 13H             ;TASTE F3
        CALL Z RBEF        ;BEFEHL REASS
        CP 15H             ;TASTE F4

        CALL Z RBYTE       ;BYTE REASS.
        CP 03H             ;BRK
        JR NZ,RE4
RE3     CALL 0C809H
        JR NZ,RE5
        LD HL,0C00EH       ;EDAS-MENU
        EX (SP),HL
        RET                ;ENDE BEI BRK
RE4     CALL 0C609H
        LD HL,0B7A0H       ;KURSORPOS
        LD (0B048H),HL
        LD A,(0B04AH)
        OR A
        JR Z,RE2           ;WEITER
RE5     CALL 0C755H
        JR RE1             ;RE1
;
;UP RBEF:EINEN BEFEHL REASSEMBLIEREN
;BZW  HEAD REASS
;
RBEF    PUSH IY            ;UP REASSEMBL
        POP HL
        LD A,(ARG5)
        CP (HL)
        JR NZ,RBEF1
        INC HL
        CP (HL)
        JR NZ,RBEF1
        LD A,(ARG5+2)
        CP 01H
        JR NZ,RBEF1
        CALL PROA
        CALL AKTADR
        XOR A
        RET
RBEF1   LD DE,(ARG5-4)     ;OFFSET -> DE
        LD A,09H
        CALL 0D727H
        CALL AKTADR
        XOR A
        RET
;.
;.......................................
;UP PROA : PROLOGBYTE,NAMEN,EPILOGBYTE
;SCHREIBEN
;
PROA    LD A,(ARG5)        ;PROLOGBYTES
        PUSH AF            ; SCHREIBEN
        CALL PV1
        DEFB 23H
        DEFB 09H
        DEFM 'DEFW 0'
        DEFB 0H
        POP AF
        LD H,A
        LD L,A
        CALL PV1
        DEFB 1AH
        CALL PV1
        DEFB 23H
        DEFB 08H
        DEFB 48H
        DEFB 0AH
        DEFB 0DH
        DEFB 09H
        DEFB 00H
        INC IY
        INC IY
PROAN   CALL PV1           ;NAMEN SCHR.
        DEFB 23H
        DEFM 'DEFM
        DEFB 27H           ;'''
        DEFB 00H
PROAN1  LD A,(IY+0)
        CP 01H             ;EPILOG 01H
        JR Z,PROAE
        CP 00H             ;EPILOG 00H
        JR Z,PROAE
        CALL PV1
        DEFB 24H
        INC IY
        JR PROAN1          ;PROAN1
PROAE   CALL PV1           ;EPILOG 01H
        DEFB 23H
        DEFB 27H
        DEFB 0AH
        DEFB 0DH
        DEFB 09H
        DEFM 'DEFB 01H'
        DEFB 0AH
        DEFB 0DH
        DEFB 00H
        INC IY
        RET
;
;---------------------------------------
;BYTEWEISES REASSEMBLIEREN VON DATEN
;MITTELS TASTE F4
RBYTE   CALL PV1
        DEFB 023H
        DEFB 09H
        DEFM 'DEFB 0'
        DEFB 000H
        LD A,(IY+0)

        CALL PV1
        DEFB 01CH
        CALL PV1
        DEFB 023H
        DEFM 'H'
        DEFW 00909H
        DEFM ';'
        DEFB 000H
        LD A,(IY+0)
        CP 020H
        JR C,BEND          ;STEUERZEICHEN
        CP 07FH
        JR NC,BEND         ;KEINE ZI,ZEI,BU
        CALL PV1
        DEFB 024H          ;ASCII-ZEICHEN
BEND    CALL PV1
        DEFB 023H
        DEFW 00A0DH        ;NL,CR
        DEFB 000H
        INC IY
        CALL AKTADR
        XOR A
        RET
;
;EDASTAST (REASSINIT)BELEGT DIE
;FUNKTIONSTASTEN F1 BIS FC
;DIE SHIFT F-TASTEN SIND FUER
;LISTENFORMATIERUNG VORGESEHEN
;(EIN,AUS,SEITENVORSCHUB,STOP FUER
;EINZELBLATTDRUCK
;DAS FOLGENDE PROGRAMMTEIL WURDE
;MIT REASS ERSTELLT UND DURCH
;KOMMENTARE ERGAENZT
        DEFW 0D0D0H
        DEFM 'EDASTAST' ;(REASSINIT)
        DEFB 01H
        LD BC,00025H
        LD HL,TABI
        LD DE,0B900H
        LDIR
        RET
TABI    DEFB 000H          ;
        DEFB 005H          ;EDAS F1
        DEFB 000H          ;
        DEFB 006H          ;EDAS F2
        DEFB 000H          ;
        DEFB 013H          ;REASS F3
        DEFB 000H          ;
        DEFB 015H          ;REASS F4
        DEFB 000H          ;
        DEFB 009H          ;F5(BEI BEDARF)
        DEFB 044H          ;D
        DEFB 045H          ;E
        DEFB 046H          ;F
        DEFB 057H          ;W
        DEFB 020H          ;
        DEFB 030H          ;0
        DEFB 000H          ;
        DEFB 009H          ;F6(BEI BEDARF)
        DEFB 044H          ;D
        DEFB 045H          ;E
        DEFB 046H          ;F
        DEFB 042H          ;B
        DEFB 020H          ;
        DEFB 030H          ;0
        DEFB 000H          ;
        DEFB 08FH          ;SH F1 EIN
        DEFB 000H          ;
        DEFB 09CH          ;SH F2 AUS
        DEFB 000H          ;
        DEFB 08CH          ;SH F3 FORM FEED
        DEFB 000H          ;
        DEFB 093H          ;SH F4 STOP
        DEFB 000H          ;
        DEFB 042H          ;B(FREI BELEGB.)
        DEFB 000H          ;
        DEFB 043H          ;C(FREI BELEGB.)
        DEFB 000H          ;

;           R E A S S      VERS.2.1
; ERLAUBT DIREKTES BEFEHLS-BZW.BYTE-
; WEISES REASSEMBLIEREN IN )EDAS(
; DES MODULS M027 DEVELOPMENT.
; REASS IST VOM QUELLTEXT AUF BELIEBIGE
; ADRESSBEREICHE )BA00H  LADBAR.
; NACH AUFRUF ERLAUBT DAS PROGRAMM ALLE
; IM EDIT-MODUS GUELTIGEN FUNKTIONEN,
; MITTELS TASTE F3 KANN BEFEHLSWEISE
; REASSEMBLIERT WERDEN.
; MITTELS TASTE F4 ERFOLGT EINE BYTE-
; WEISE RUECKUEBERSETUNG VON DATEN.
; DRUCKBARE ZEICHEN WERDEN ALS KOMMEN-
; TAR MIT ABGELEGT.
; (F3=13 =STOPTASTE , F4=15H)
; EDASTAST  BELEGT DIE FUNKTIONSTASTEN
; 1 BIS 12
; (05/06/13/15/DEFW 0/DEFB 0)
; (8FH,9CH,8CH,93H,''B'',''C'')
; VOR EINGABE IN DIE LETZTE ZEILE MIT
; SHIFT/CURSOR DOWN UMBLAETTERN.
; BEI FEHLENDEM ARG ERFOLGT RUECKSPRUNG
;
; ZUEHLSDORFF VEB NACHRICHTENELEKTRONIK
; ARNSTADT BETRIEBSBERUFSCHULE
; AUGUST 1987
;
```

Ein FORTH-83 basiertes 16-Bit-Entwicklungssystem

„NILES-FORTH" ist ein modular organisiertes FORTH-83-basiertes Programmentwicklungssystem für 16-Bit-Mikrorechner (auch als Entwicklungssystem für Fremdhardware einsetzbar). Versionen für MSDOS, CP/M-86, CP/M-80, BOS 1834, SCPX, SCP u. a. sind verfügbar.
Das System unterstützt das MSDOS-Handle-Konzept bzw. bildet dieses in den anderen Versionen nach.
Es enthält einen sehr konfortablen **Screeneditor**, der u. a. über Blockfunktionen (Einfügen/Löschen/Splitten), einen Zeilenstack, direkte, definitionsorientierte oder indexzeilenorientierte Blockanwahl, Speicherung von Editorpositionen, eine automatische Signierfunktion und einen eigenen menügestützten File- und Directory-Service verfügt. Er unterstützt die Arbeit mit Schattenblöcken, ist einfach zu bedienen (HELP-Funktionen) und benutzt installierbare Funktionstasten, deren Funktionen in Menüleisten angezeigt werden. Bildschirm- und Tastaturtypen sind installierbar (für IBM-Kompatible BIOS-Versionen direkte – sehr schnelle – Terminal-Ein-/Ausgabe).
Der **Debugger** ermöglicht dem Nutzer die verschiedensten Testmodi von High-level-Definitionen (Schritt-, Lauftest, Trace) mit ausführlicher Statusanzeige und mehreren zusätzlichen Diensten (u. a. direkte Änderung von compilierten Definitionen, Aufruf des FORTH-Interpreters während des Tests). Der **Decompiler** überträgt ausführbaren FORTH-Code zurück in Quellcode mit gleichzeitiger Ausgabe der Speicherbelegung. Mittels des **Metacompilers** können fertige FORTH-Programme in eine endgültige (optimierte) Version gebracht sowie FORTH-Kernsysteme für andere Mikroprozessoren und Betriebssysteme bzw. Systeme mit modifizierten Eigenschaften erzeugt werden.
Zusatzmodule: Arbeit mit freiformatigen Dateien (listen, laden, konvertieren), automatische Such- und Listfunktion für Glossaries (WS-kompatibel), komfortable Stringeingabe und Zeichenkettenfunktionen, Druckprogramme u. v. a.

VEB WMK „7. Oktober", Abt. EAR/B, Gehringstr. 39, Berlin, 1120; Tel. 3631641

Karadshow

Neue Netzwerkkonzeption

Im Zusammenhang mit einer eigenen Rechnerentwicklung und dem Aufbau eines zweiten Computerkabinetts an der ESOS Kleinmachnow wurden die Netzwerkkonzeption BASnet (BASware Network) und die Übertragungsprozedur SCALP (Self controlling advanced linkage protocol) entwickelt. Gegenüber den in der Literatur bekannten Lösungen zeichnen sie sich durch die Anpassungsfähigkeit an beliebige Datenendplätze aus, was eine Selbstoptimierung des Systems hinsichtlich der Datenübertragungsrate ermöglicht.
Da eine serielle Übertragung gewählt wurde (das Prinzip ist auch auf parallele Netze anwendbar), kommt das Netz mit 3 bis 4 Leitungen aus. Die Struktur ist busförmig, es existiert jedoch ein ausgezeichneter Busnutzer (driver). Als benötigte Hardware sind nur die üblichen U880-Schaltkreise in Verbindung mit einfachen Schaltungen vonnöten.
BASnet ermöglicht durch seine Struktur den gleichzeitigen Anschluß von Computern beliebiger Leistungsfähigkeit. Damit ist es für innerbetriebliche Netze gut geeignet. Die Datenrate kann in beliebigen Bereichen mit Hilfe von SCALP festgelegt werden, je nach den Erfordernissen und Möglichkeiten des Gesamtsystems.
Die SCALP-Software ist im Prinzip auch in höheren Programmiersprachen zu erstellen, was eine schnelle Anpassung von neuen Computern ohne große Programmierfertigkeiten ermöglicht. Sie kann jedoch auch für die meisten Rechner von BASware in Assembler erstellt werden, was die Datenrate erhöht.

ESOS „Georg Thiele", Am Weinberg 20, Kleinmachnow, 1532

Düring

GEDIT-M86-kompatibles Grafiksystem für FORTRAN77 und TURBO-PASCAL

An der Sektion Schiffstechnik der Wilhelm-Pieck-Universität Rostock wurde ein Grafiksystem für die Sprachen FORTRAN77 und TURBO-PASCAL zur effektiven Gestaltung der grafischen Arbeit am Arbeitsplatzcomputer AC 7100 entwickelt. Es enthält für beide Sprachen gleichlautende Unterprogramm- bzw. Prozeduraufrufe und lehnt sich in der Syntax und im Funktionsumfang an die auf dem HP 9845B realisierten Grafikbefehle an. Der Nutzer benötigt bei der Anwendung dieses Grafiksystems keine Kenntnisse über pixelorientierte Bildschirmkoordinaten. Darüber hinaus ist das Grafiksystem kompatibel mit dem grafischen Editor GEDIT-M86, das heißt, mit seiner Hilfe auf Diskette erzeugte Bilddateien können mit GEDIT-M86 bearbeitet und mit einem Importbefehl vom Grafiksystem wieder gelesen werden.
Die Grafikbefehle für FORTRAN77 liegen in Form einer Objektmodulbibliothek vor. Innerhalb der vom Anwender geschriebenen FORTRAN77-Programme werden sie durch Unterprogrammaufrufe aktiviert.
Die Grafikbefehle für TURBO-PASCAL liegen in Form einer Quelltextbibliothek vor. Innerhalb der vom Anwender geschriebenen TURBO-PASCAL-Programme werden die entsprechenden Grafikbefehle durch Prozeduraufrufe aktiviert. Die jeweils benötigten Grafikprozeduren sind daher durch INCLUDE-Anweisungen in das Programm einzubinden. Hierfür steht eine Datei zur Verfügung, die beim Editieren an den Anfang des Nutzerprogramms gelesen wird. Sie enthält die INCLUDE-Anweisungen für alle verfügbaren Grafikprozeduren. Der Nutzer hat dann die Möglichkeit, die nicht benötigten Prozeduren von der Einbindung in sein Programm auszuschließen, z. B. durch Löschen des Dollarzeichens, wodurch die INCLUDE-Anweisungen zu Kommentaren werden. Durch diese Arbeitsweise werden die Programme nicht unnötig lang und die Zeiten für die Compilierung verkürzen sich.
Die einzelnen Grafikbefehle realisieren solche Funktionen, wie das Zeichnen einer Linie, eines Kreises, eines Rechtecks, eines beschrifteten Koordinatensystems, eines Koordinatennetzes oder eines Markers, das Festlegen der entsprechenden Linienstärken, -typen und -farben, den Export und Import von Bildern auf bzw. von Diskette u.v.a.m. Dabei kann gleichzeitig und unabhängig voneinander in bis zu 10 Fenstern gearbeitet werden, die bei Bedarf auch überlappen können.

Wilhelm-Pieck-Universität Rostock, Sektion Schiffstechnik, Sonderforschungsbereich Theoretische und experimentelle Hydromechanik, Albert-Einstein-Str. 2, ROSTOCK, 2500

Dr. A. Stern

Tuschezeichenstifte für HP-Plotter 7580 . . . 86

Zur Import-Einsparung wurden für HP-Plotter 758X anstatt der originalen Langkörperzeichenstifte die lin's CAD-Zeichenspitzen vom VEB Heiko Wernigerode nach entsprechenden Anpaßarbeiten eingesetzt. Die Anpaßarbeiten beziehen sich auf die Mundstücke der normalen Tuschezeichengeräte von Heiko lin's 9 Plus für die herkömmliche Zeichentusche bzw. von lin's 9 Plus F für folienlösende Tusche. Sie wurden so bearbeitet, daß sie in die Plotteradapter passen.
Die so angepaßten Handzeichenmundstücke nehmen dann ohne weitere Maßnahmen die Plotterzeichenspitzen auf. Diese sind auch so gestaltet, daß sie im HP-Adapter zentriert sind. So tritt beim Plotten mit verschiedenen Strichstärken kein Versatz zwischen den Konturen auf.

VEB Werkzeugmaschinenfabrik „Hermann Matern", PSF 59, Mittagstraße 16, Magdeburg, 3018

Krause/Hahn

Erleichterte Erstellung von Dialogmasken für PASCAL-Programm

Von uns wurde ein Programmgenerator für die Erstellung von Bildschirm-Dialogmasken in PASCAL entwickelt. Er ist nutzbar für die Rechnertypen PC 1715 und AC A 7100.
Der Kursor läßt sich frei über den Bildschirm verschieben. Unter Nutzung der Funktionstasten können folgende Kommandomodi eingestellt werden:
- Texteingabe
- Setzen/Löschen von Feldbegrenzungspunkten
- Zeichnen/Löschen von horizontalen bzw. vertikalen Linien zwischen Feldbegrenzungspunkten
- Zeichnen/Löschen von Rechtecken zwischen den Begrenzungspunkten
- Löschen des Fensters zwischen den Begrenzungspunkten
- Kopieren des Fensters
- Beschleunigte Kursorbewegung
- Festlegung der Eingabefeldgrößen
- Helpfunktion.

Korrekturen sind in der Erstellungsphase unbegrenzt möglich. Das Bildschirmbild läßt sich als Kopie abspeichern und für weitere Bearbeitungen wieder laden. Aus der Kopie wird eine PASCAL-Prozedur mit vorgebbaren Namen generiert. Dabei werden die Ausgaben als write-Anweisungen und die Eingaben in Form einer Pseudoprozedur „input(xposition,yposition,länge);" realisiert.
Effekte:
- Wesentliche Arbeitszeiteinsparung bei der Programmierung von Eingabedialogen
- Mehrfachnutzung von Entwürfen innerhalb eines Programmsystems.

VEB Universal Dresden, BT Kunststoffverarbeitung Langenhennersdorf, OSF, Pirna 6, 8301, Tel. Langenhennersdorf bei Pirna 217

Graf/Dr. Kurz

Dateivewaltungsprogramm ELGE

Für das Elektrogewerk liegt das Programm ELGE für Handwerksbetriebe und PGH vor. Grundlage ist die PAO 564. ELGE ist in dBASE geschrieben und arbeitet auf Rechnern mit mindestens zwei Laufwerken (A 5120/30, PC 1715, A 7100, C 128 usw.; Mindestkapazität je Diskette 340 KByte). Von der Auftragsannahme bis zur Rechnungslegung, für Betriebe und Bürger, verwaltet das Programm die PAO sowie die Dateien für Aufträge, Kunden, Rechnungen, Bestände usw. Bei einem Materialzugang bzw. der Rechnungslegung werden die Bestandsdateien (Lagerfachdateien) aktualisiert. Die Rechnungslegung erfolgt automatisch nach dem Aufmaß (keine Eingabe von Preisen notwendig). Ebenso wird der Verrechnungsbetrag zur Rückfinanzierung für die Finanzrechnung gespeichert. Alle Dateien, die Rechnungen sowie die Inventurlisten lassen sich beliebig oft ausdrukken. Der Lagerbestand in Mark steht zu jeder Zeit zur Verfügung. Die Koeffizienten der PAO werden nach einer Korrektur automatisch eingearbeitet. Neue Preislisten werden selbsttätig gebildet und stehen zur Eingabe bereit. Ein geringer Bedienaufwand wird durch klare Menüführung und automatische Dateiverwaltung erreicht. Eine Fehlererkennung sowie ein komfortabler Korrektur- und Eingabemodus stellen keine Anforderungen an den Bediener in bezug auf Programmkenntnisse.

Elektroinstallation Fa. Ulrich Schmidt, Bahnhofstr. 17, Altlandsberg, 1274; Tel. 408

Schmidt

Textverarbeitung und Kassettensteuerung für KC 87

Für den Kleincomputer KC 85/1 bzw. KC 87 wird ein Textverarbeitungssystem angeboten, welches Eingabe, Korrektur, Druck und Archivierung beliebiger Texte ermöglicht. Das als

Fullscreeneditor implementierte System erlaubt mittels komfortabler Bildschirmarbeit die Eingabe von Text über Tastatur und Kassette, das Streichen und Einfügen von Zeilen, Zeichen und ganzen Textabschnitten sowie die Erzeugung der Rechtsbündigkeit des Textes. Es stehen Funktionen zum Verschieben und Kopieren ganzer Textabschnitte zur Verfügung. Da der Bildschirm des KCs nur 40 Spalten/Zeile darstellen kann, wird bei Erreichen des Zeilenendes der gesamte Bildinhalt nach links geblättert, so daß auch Tabellen komfortabel erarbeitet werden können. Die Bedienung des Textverarbeitungssystems erfolgt über die Computertastatur oder die Tastatur einer angeschlossenen Schreibmaschine. Das Textverarbeitungssystem und der Schreibmaschinenanschluß sind im Technikum LAURA nachnutzbar.
Weiterhin wurde ein steuerbares Kassettengerät an den KC 85/1 angeschlossen. Dieser Anschluß erlaubt eine inhaltsverzeichnisorientierte Dateiarbeit auf der Kassette, wie sie sonst nur bei angeschlossenen Diskettenlaufwerken üblich ist. Das Inhaltsverzeichnis wird am Anfang der Kassette abgespeichert, so daß alle Programme die von ihnen benötigten Dateien selbständig auf dem Band suchen und Laden bzw. diese an freien Stellen auf dem Band abspeichern. Die Hard- und Software zur Ansteuerung eines Kassettengerätes vom Typ SK 3000 kann im Technikum nachgenutzt werden. Weiterhin gehören zum Nachnutzungsumfang komfortable Kassettenkomprimier- und Kopierprogramme.

Friedrich-Schiller-Universität Jena, Sektion Technologie für den WGB, Technikum LAURA, Ernst-Thälmann-Ring 32, Jena, 6900; Tel. 8222132

Prof. Dr. G. Entreß

Programm zur Einstellung des Druckers FX 1000 durch den MR A 7100

Die modernen Nadeldrucker können vom Rechner her durch eine Fülle von Steuerfolgen sehr flexibel eingestellt und eingesetzt werden. Programmierbar sind die Zeichen- und Zeilenabstände, die Schriftgröße, die Schriftqualität, die Schriftart, der Zeichensatz und weitere Druckerfunktionen.
Für die Aktivierung dieser Funktionen wurde für den A 7100 ein Maschinenprogramm (2 KByte) erarbeitet, das bei der Einstellung des Druckers FX 1000 eine unentbehrliche Hilfe darstellt.
Das Programm umfaßt 3 Betriebsarten:
1. Durch 59 Eintastenbefehle (Groß- und Kleinbuchstaben, Ziffern) können alle wesentlichen Druckerfunktionen rasch und bequem aktiviert und deaktiviert werden. Jeder Befehl wird durch einen kurzen Bildschirmtext quittiert.
2. Einzelworte und Kurztexte (max. 255 Zeilen) können über die Tastatur eingegeben, auf dem Bildschirm kontrolliert und dann entsprechend der aktuellen Druckereinstellung ausgedruckt werden. Auch die Ausgabe von Steuerfolgen, soweit sie von der Betriebsart 1 nicht erfaßt werden, ist auf diese Weise möglich. Steuerfolgen und Texte dürfen gemischt auftreten.
3. Die Definition einzelner nutzereigener Drucksymbole wird unterstützt. In bequemer Weise sind pro Druckzeichen 13 Bytes (als je 2 Hexzeichen) über die Tastatur einzugeben.

Akademie der Wissenschaften der DDR, Zentralinstitut für Astrophysik, Sternwarte Sonneberg, Postfach 55–27, Sonneberg, 6400; Tel. 2287.

Dr. W. Fürtig

Ein universelles CAQ-System zur Optimierung, Schwachstellenanalyse und Qualitätssicherung komplexer Prozesse

Die Analyse und Modellbildung komplexer Probleme und Prozesse war bisher ausschließlich eine Domäne rechentechnisch gut bestückter wissenschaftlicher Zentren. Die Entwicklung der Personalcomputertechnik und effizienter Basissoftware bieten die Grundlage dafür, entsprechende Methoden auch dem informationstechnisch und mathematisch weniger vorgebildeten Anwender (Verfahrenstechniker, Technologen, u. a. Fachingenieure) zugänglich zu machen.
Die notwendige Aufbereitung leistungsfähiger Verfahren der modernen Prozeßanalyse mit speziell entwickelten, technologisch zielorientierten Bewertungskriterien und nachgeschaltetem Beratungssystem erfolgte in Form des universell anwendbaren Analyse- und Beratungssystems MAVEX.
Robuste Methoden und Lernfähigkeit garantieren Modelloptimierung und -nachführung auch bei stark gestörten und nichtlinearen Systemen. Umfangreiche Dialogroutinen für Kommandointerpretation, Textverarbeitung, Datenbankorganisation, Fehlerbehandlung, Grafik und Druck in einem hierarchisch aufgebauten Systemkonzept realisieren eine anwenderfreundliche und überschaubare Bedienoberfläche. Es findet Anwendung in allen Bereichen der verfahrens- und verarbeitungstechnischen Industrie zur effektiven Analyse komplizierter Prozesse und unübersichtlicher Datenmengen mit dem Ziel, ein der Steuerungsaufgabe genügendes mathematisches Modell zu finden bzw. Steuerentscheidungen und Führungsgrößen zu objektivieren.
Das System ist gegenwärtig unter allen CP/M- und PC/MS-DOS-kompatiblen Betriebssysteme implementierbar.
Die Anwenderdokumentation besteht aus
– Zusammenfassung
– Problemstellungen potentieller Anwendungsbereiche
– Mittel und Methoden
– Systemkonzept und Vorgehensweise
– Leistungsangebot
– Beispiel
– Erläuterung der Bedieneroberfläche
Sie kann unverbindlich angefordert werden.

Ingenieurhochschule Cottbus, Sektion Technologie der Bauproduktion, Wissenschaftsbereich I, PSF 102, Cottbus, 7500; Tel. 690 App. 2860

Prof. Dr. Unruh

Schlüsselhandbuch

Im VEB Kombinat Mechanisierung Karl-Marx-Stadt wurde ein für alle Kombinatsbetriebe verbindliches Schlüsselhandbuch entwickelt.
Dieses Handbuch dient den Betrieben des Kombinates zur Erhöhung ihres Organisationsniveaus sowie zur Nutzung von einheitlichen Datenverarbeitungslösungen.
Es hat sich in der Praxis hervorragend bewährt. Dies zeigt die Anwendung im eigenen Kombinat und darüber hinaus der Vertrieb an zirka 50 Betriebe aus allen Gebieten der DDR.
Die im Jahre 1986 erarbeitete Fassung wurde 1987 um einige Schlüssel erweitert.
Der Inhalt der angeführten Schlüssel widerspiegelt alle wesentlichen Phasen des Reproduktionsprozesses, ist von Betrieben mit unterschiedlichem Produktionsprofil anwendbar und entspricht den gesetzlichen Bestimmungen.
Der Prozeß der Anwendung der Schlüssel kann im Zusammenhang mit der Einführung von EDV-Projekten bzw. in Verbindung mit dem Aufbau von Primärorganisationslösungen mit vertretbarem Aufwand bewältigt werden.
Da Schlüsselhandbuch stellt somit eine solide Grundlage für die Realisierung laufender und künftiger Arbeiten dar.
Das Schlüsselhandbuch liegt als Textdatei vor. Die Übergabe wird über Standard- oder Minidiskette realisiert.

VEB Organisation und Rechentechnik Karl-Marx-Stadt, Abt. Betriebsberatung, PSF 743, Karl-Marx-Stadt, 9010; Tel. 32004

R. Müller

Programmkomplex Energieabrechnung

Hardware: beliebiger PC- oder BC-Typ; beliebiger Druckertyp
Software: SCP oder kompatibles System
Programmiersprache: TURBO-PASCAL
Programmsystem:
Das Programmsystem ist darauf ausgerichtet, die gesamten Energieabrechnungen und -auswertungen zu unterstützen bzw. zu übernehmen. Da es sich um ein Modell handelt, ist der Komplex in Umfang und Leistungsfähigkeit vom Nutzer den individuellen Anforderungen anpaßbar.
Es lassen sich Summen- und Mischungsverhältnisse über beliebige Energieträger bilden. Alle Energieträger sind mit Bezeichnung, Preis/Mengeneinheit, Heizwert/Mengeneinheit, Mengeneinheit und Summendarstellung frei vereinbar.
Für Transport und Umlagerungen werden Zahlungsanweisungen als Beleg erstellt.
Sonstige Auswertungslisten:
Energieabrechnung, Kosten, Soll, Zugang.
Die Listen werden über beliebige Zeiträume wahlweise für einen Bereich oder den gesamten Betrieb erstellt.
Es existiert ein zusätzlicher „Bereich", der in keine Summe einfließt und zur Bearbeitung vom Formblatt 121/40 dient. Status listet alle installierten Daten und Bezeichnungen.

Schwarzer

PASCAL/BASIC-Maskengenerator (PBM)

Der PBM dient dem BASIC- oder PASCAL-Programmierer als Programmierhilfe. Mit ihm können beliebige Textfiles (Masken) in BASIC- oder PASCAL-Quelltexte übersetzt werden. Diese Masken dürfen Ein- und Ausgaben von Variablen beinhalten. Mit dem im PBM integrierten Editor können Masken bis zu einer Größe von 255 Spalten und 80 Zeilen erstellt werden. Die in der Maske ein- bzw. auszugebenden Variablen werden durch Steuerzeichen gekennzeichnet. Der PBM übersetzt diese Masken in Programme zur Ausgabe auf Drucker oder auf Bildschirm. Lauffähig ist der PBM auf dem PC 1715 und auf dem BC A 5120 unter einem CP/M-kompatiblen Betriebssystem.

Ingenieurhochschule Berlin, Prorektorat Naturwissenschaft und Technik, Marktstr. 9, Berlin 1134

St. Kaiser

Wir suchen ...

... eine Möglichkeit, die Software des PC 1715, BC 5120 (5130) und K 8924 (eventuell auch A 7100) auf einem EC 1834 nutzen zu können.

VEB (K) Bau Tribsees, Projektierungsgruppe, Straße der Befreiung 65, Stralsund, 2300

Engelbrecht

... für den AC 7150 (Betriebssystem DCP 3.2) lauffähige Softwarelösungen für Finanzrechnung/Kontokorrent und Lohnrechnung.

VEB Plastverarbeitungswerk, Hauptbuchhalter, Schwerin-Sacktannen, 2767; Tel. 452535 oder 452243

Regner

... Softwarelösungen für PC 1715 zur Thematik:
1. Energieplanung und -abrechnung auf den Vordrucken 1910, 1911, 1912, 1915, 1916, 1917, 1918, 1919, 1920, 1972 sowie S 121-j, S 121-3, S 121-40, S 121-12.
2. Innerbetriebliche Energieabrechnung und -verteilung auf Kostenstellen für Industriekraftwerke, insbesondere für die Energieträger feste Brennstoffe, Dampf (Erzeugung, Abgabe, Verbrauch), Kondensat, Elektroenergie (Erzeugung, Verbrauch, Abgabe, Bezug) sowie Hilfsmaterial wie Kalk, Salz usw.

VEB Waschmittelwerk Genthin, Postfach 43, Hilde-Coppi-Str. 56, Genthin, 3280

Köhler/Seidel



Literatur

Lokale Netze

von H. Löffler, Akademie-Verlag, Berlin, 1987, 245 Seiten

Das vorliegende Fachbuch stellt eine grundlegende Einführung in das Gebiet der lokalen Rechnernetze (Local Area Network – LAN)dar, wobei der Autor bemüht war, die gesamte Breite dieses noch relativ jungen Fachgebietes widerzuspiegeln.
Im ersten Abschnitt wird deshalb eine gut gelungene Einführung in Aufbau und Wirkungsweise lokaler Netze gegeben, um den Leser vor allem in die Begriffswelt einzuführen und ein Überblickswissen zu vermitteln.
Im zweiten Abschnitt wird eine umfassende Analyse der Kanalzugriffsmethoden vorgenommen, die in den siebziger Jahren unter Benutzung von Funkkanälen eingesetzt wurden und den Ausgangspunkt der Entwicklung des Spezialgebietes der lokalen Netze bildeten.
Den Schwerpunkt des Buches mit 40 Prozent des Umfangs bildet der Bedeutung entsprechend der dritte Hauptabschnitt zu lokalen Netzen mit linienförmiger Struktur (Linienbusse). Neben einer Vielzahl mehr oder weniger bekannter Verfahren werden ausführlich das Ethernet- (CSMA/CD) und das Tokenbus-Verfahren dargestellt.
Im vierten Abschnitt wird auf lokale Netze mit ringförmiger Struktur und im fünften Abschnitt auf die Verkopplung von Netzen über sogenannte Gateways eingegangen.
Den Schlußteil des Buches bilden eine sehr nützliche Übersicht über Standards zu lokalen Netzen, ein umfangreiches Literaturverzeichnis sowie Symbol-, Abkürzungs- und Sachwortverzeichnisse.
Charakteristisch für das Buch ist, daß der Autor versucht hat, alle Aussagen durch Beispiele zu real vorhandenen und funktionierenden Netzen zu untersetzen, wodurch das Buch zweifellos eine gewisse Breite bekommt. Bei der stürmischen Entwicklung des Gebietes der lokalen Netze in den letzten 10 Jahren war jedoch bei dem bescheidenen Umfang von nur 245 Seiten eine umfassende Monografie nicht möglich. Vielleicht ist eine Monografie auch nicht erforderlich, denn was für die Informatikausbildung und Praxis benötigt wird, ist ein gut ausgewogenes Standardwerk. Ich bin überzeugt, daß das Buch von Löffler diesen Anspruch erfüllt.

Prof. Dr. Thomas Horn

Daten integrierter Schaltkreise

(CMOS-Schaltkreise)
von K. Streng, Militärverlag der DDR, Berlin 1987, 192 Seiten, 314 Bilder, 7,– M

Nach den „Transistordaten", „Diodendaten", „Daten linearer integrierter Schaltkreise (I)", „Daten digitaler integrierter Schaltkreise – TTL-Schaltkreise –" liegt nunmehr der dritte Teil der „IS-Datenbücher" vor, die „Daten digitaler integrierter Schaltkreise – CMOS-Schaltkreise –".
Als folgerichtige Fortsetzung der Datenbuch-Reihe und insbesondere der „TTL-Schaltkreise" ist der vorliegende Band nach gleichen Gesichtspunkten aufgebaut.
Da die CMOS-Schaltkreise auf Grund einiger entscheidender Vorteile langsam die bekannten TTL-Schaltkreise verdrängen, ist der aus dem TTL-Schaltkreis-Buch übernommenen Gliederung ein Abschnitt 1.1 vorangestellt. Dieser geht kurz auf die Besonderheiten der CMOS-Technologie ein. Dabei setzt aber diese Erklärung beim Leser – oder genauer gesagt Benutzer – dieses Buches bereits ein gewisses Vertrautsein mit CMOS voraus.
Mit größtmöglicher Vollständigkeit gibt dieser Band wiederum dem praktisch tätigen Elektroniker in gedrängter, aber dennoch informationsreicher Darstellungsweise eine große Datenmenge in die Hand.
Wie schon in den vorhergehenden Büchern praktiziert, beschränkt sich auch dieser Band auf die Standard-CMOS-Reihe und bleibt seinem Charakter als Datenbuch treu, indem aus verständlichen Gründen Anwendungsbeispiele fehlen.
Über das umfangreiche Typenverzeichnis findet der Benutzer zum gesuchten Schaltkreis jeweils Daten, die zugehörige Anschlußbelegung und für die meisten Typen eine Wahrheitstabelle.
Diese Informationen sollten den erfahrenen Amateur und überhaupt den praktisch Tätigen in die Lage versetzen, sich der CMOS-Schaltkreis-Reihe zu bedienen.

Dr. I. Schreiber

KC-Anwenderkatalog

Mitte Juli dieses Jahres erschien die 2., überarbeitete, Auflage des Anwender-Softwarekatalogs, der von der Applikationsstelle des VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen herausgegeben wird. In dem Katalog finden erstmalig neben vielen Softwarelösungen auch Hardwarelösungen ihren Platz. Des weiteren beinhaltet die 31 Seiten umfassende Broschüre eine Übersicht von Programmkassetten des Mühlhausener Kleincomputerherstellers, die noch in diesem Jahr im Einzelhandel angeboten werden.
Die Applikationsstelle wird auch künftig Programme von KC-Anwendern erfassen, um jährlich einen aktuellen Stand des Softwarekatalogs zu gewährleisten.
Der Katalog ist ab sofort kostenlos in der Applikationsstelle oder im Bereich Beschaffung und Absatz des VEB Mikroelektronik „Wilhelm Pieck" Mühlhausen, Eisenacher Str. 40, Mühlhausen, 5700, zu beziehen.

M. Göring

Zanimatel'naja matematika i personal'nyj komp'juter (Unterhaltsame Mathematik und Personal Computer)

von Czes Kosniowski, Übersetzung aus dem Englischen, 1. Auflage. Moskva: Mir 1987, 192 Seiten, Paperback, 4,– M

Der Inhalt der Broschüre hält, was ihr Titel verspricht. In unterhaltsamer und aufgelockerter Form wird der Leser mit Hilfe der – besonders für Jugendliche sehr attraktiven – Computertechnik an verschiedene Gebiete der Mathematik herangeführt. Dies betrifft unter anderem Folgen und Reihen, Matrizenrechnung und Geometrie ebenso wie Graphiken, Differentialgleichungen, Funktionen mit zwei Veränderlichen oder die Spieltheorie.
Das Buch ist jedoch auch aus anderer Sicht interessant. Mit der Entwicklung der Mikrorechentechnik erfuhr vor allem die Programmiersprache BASIC eine weite Verbreitung. Zu allen Mathematikspielen wurden daher entsprechende BASIC-Programme angegeben. Diese entstanden in der Fakultät für Rechentechnik und Informatik der Universität Newcastle upon Tyne. Sie wurden für die sowjetische Ausgabe – meistenteils mit russischsprachigen Kommentaren und Ausschriften versehen – auf Rechnern des Typs DVK-2M getestet. Eine zusätzliche Bereicherung dieser interessanten Broschüre stellen die zahlreichen Graphikbeispiele und Bildschirmausdrucke dar. Es sei aber betont, daß die Rechentechnik hier vor allem als Mittel zum Erlernen der Mathematik verstanden wird. Nichtsdestoweniger bietet das Buch für alle Computerfreunde eine Fülle von Ideen und fertigen Programmen. Dabei wurden allgemein verbreitete BASIC-Kommandos verwendet, die weitestgehend auf spezielle Funktionen verzichten. Somit dürfte die Implementierung dieser Software auf Kleinrechnern der DDR-Produktion kein Problem darstellen.
Zielgruppe dieses unterhaltsamen und nützlichen Buches sind vor allem an Informatik und Rechentechnik interessierte Oberschüler und Studenten, aber auch alle anderen PC-Einsteiger. Nicht nur hinsichtlich ihres Inhaltes, sondern auch in ihrer Aufmachung und Druckqualität kann diese reich illustrierte Broschüre als außerordentlich gelungen bezeichnet werden.

G. Mau

Basic: Lösung von Produktionsaufgaben

(BEJSIK: Rešenie proizvodstvennyh zadač)
von J. Niks in russischer Sprache, Übersetzung aus dem Englischen, Verlag Mašinostroenie Moskau 1987, 248 Seiten mit Bildern, Kunstleder 7,– M

Mikrocomputer werden heute auch in der Produktion zur Lösung vielfältiger Probleme genutzt. Hauptziel des Buches ist es, den Technologen und Ökonomen zu zeigen, daß Mikrocomputer eine schnelle, genaue und gut systematisierte Antwort auf Fragen des Produktionsprozesses bei minimalen Kosten zu geben in der Lage sind. Die einzelnen Kapitel des Buches sind so aufgebaut, daß zunächst das Wesen der Aufgabe analysiert wird. Dann wird ein Programmtext in BASIC angeführt, der die Lösung der Aufgabe anbietet. Anschließend werden die Programme kommentiert. Dabei wird die Kenntnis von BASIC vorausgesetzt. Am Ende jedes Kapitels werden Aufgaben zur selbständigen Lösung gestellt.
Folgende Problemkreise werden in den zwölf Kapiteln behandelt: Kostenrechnung; Grafiken für Produktionskennziffern; Analyse technologischer Varianten; Entscheidungshilfen für Investitionen; Regressionsanalyse; Arbeitszeitanalyse; grafische Darstellung von Analysen der Produktionsprozesses; Methoden der File-Verarbeitung; Projektierung technologischer Prozesse; Berechnung von Konstruktionsdaten.
Das Buch behandelt aktuelle Fragen des Produktionsprozesses im Zusammenhang mit Mikrorechnern. Es wendet sich an ingenieurtechnische Mitarbeiter und Ökonomen von Industriebetrieben. Die angegebenen Programme sind auch auf Rechnern der DDR-Produktion lauffähig. Daher ist es allen zu empfehlen, die mit solchen Problemen zu tun haben.

Dr. H. Dobesch

Personalcomputer – Einführung in Technik und Gebrauch

von D. Dietrich und H. Metzendorf. Verlag R. v. Decker & C. F. Müller Heidelberg 1987, 293 S., 16,80 DM, ISBN 3–8226–3086–1

Dieses Buch wurde geschrieben für den Einsteiger, der jedoch bereits Grundkenntnisse der Elektronik besitzen muß.
Schon die Überschriften der Kapitel – Einführung, Komponenten eines Mikrocomputers, Peripheriekomponenten, Betriebssysteme, Software-Tools, Anwendersoftware, Kommunikation mit Personalcomputern, Datensicherheit und Bewertung technischer Daten – zeigen, daß kaum ein Gebiet ausgelassen wurde.
Die zahlreichen Begriffe werden anschaulich mit vielen Abbildungen und i. a. auch recht umfassend erläutert. Sehr gut aufbereitet und informativ ist das Kapitel „Peripheriekomponenten" und hier der Abschnitt über Displays, wo Informationen über Aufbau und Funktionsprinzip gegeben werden.
Hinsichtlich der Aktualität läßt die Publikation etwas zu wünschen übrig. Beispielsweise wird das Funktionsprinzip eines Mikroprozessors an Hand des bereits recht alten 8080A erläutert. Vergeblich sucht man auch danach, was ein Koprozessor oder gar ein PC/AT (bzw. dazu kompatibler PC) ist.

I. P.



Literatur

Wirtschaftliche Software-Produktion

von B. W. Boehm
Forkel-Verlag Wiesbaden 1986
680 Seiten

Wirtschaftliche Software-Entwicklung nimmt entscheidend Einfluß auf die Effektivität der Anwendung der automatisierten Informationsverarbeitung. Modelle und Methoden für die Beherrschung wirtschaftlicher Aspekte des Software-Engineering werden durch den Verfasser in Auswertung einer Vielzahl von Projekten angeboten. Beeindruckend ist die methodisch-pädagogische Herangehensweise für Problemlösungen, gepaart mit theoretischem Niveau und praktischen Erfahrungen. Komplexe sind Gegenstand der Betrachtungen, die zweifellos auch für den Informatik-Ingenieur in der DDR von großem Interesse sind. In vier Hauptabschnitten wird versucht zu zeigen, welche Faktoren den Lifecycle eines Software-Produkts bestimmen. Fallstudien bilden unter programmtechnischen und wirtschaftlichen Aspekten den Ausgangspunkt der Betrachtungen. Der zweite Hauptabschnitt ist dem quantitativen Modell eines Software-Lifecycle gewidmet. Insbesondere wird ein von Boehm entwickeltes Construktive Cost Model (COCOMO) über mehrere Entwicklungsetappen behandelt. Der 3. Hauptabschnitt beschäftigt sich mit Leistungs- und Rentabilitätsmodellen, grundlegenden Produktionsfunktionen für die Software-Entwicklung, mit Nettowert- und Grenzwert-Analyse im Hinblick auf Software-Produkte, die 8000 bis 512000 Befehle beinhalten. Derartige Analysen und die Anwendung von Modellen waren zwangsläufig auf Grund der Vielfältigkeit der auszuwertenden Daten nur rechnergestützt praktikabel. Der vierte Hauptabschnitt enthält Prinzipien, Methoden und Verfahren der Software-Kostenschätzung und diejenigen Kostenfaktoren im einzelnen, die in das COCOMO letztlich Eingang finden und bewertet werden. Man kann beeindruckt sein von der Fülle der Fakten und Daten; Anregungen für die Weiterentwicklung angebotener Modelle und Methoden können aus der Publikation entnommen werden; es ist ein lesenswertes Buch, auch wenn keine Rezepte für die Lösung unserer Software-Probleme angeboten werden.

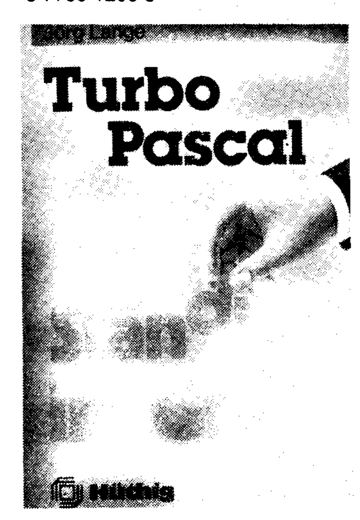
Prof. Dr. Wolfgang Schoppan

Turbo Pascal – Ständig im Griff

von J. Lange, Heidelberg: Dr. Alfred Hüthig Verlag 1987, 130 Seiten, ISBN 3-7785-1313-3

CP/M – Ständig im Griff

von R. Weiß, Heidelberg: Dr. Alfred Hüthig Verlag 1986, 87 Seiten, ISBN 3-7785-1266-8



Man muß die nötigsten Fakten über CP/M oder TURBO-PASCAL nicht in dicken Ordnern suchen, wenn man diese kleinen Referenzhandbücher in der praktischen Spiralheftung besitzt. *„CP/M ständig im Griff"* ist zum größten Teil so gegliedert, daß die Beschreibung eines neuen Befehls auf einer neuen Seite beginnt. Da die 8-Bit-Technik sicherlich noch eine ganze Weile „Dauerbrenner" bleibt, wird auch dieses Betriebssystem noch eine Weile leben. Die Erläuterungen zu den Befehlen gehen auf eventuelle Unterschiede zwischen den Betriebssystemen CP/M-80, CP/M-86 und CP/M-Plus (3.0) ein. ED, der Editor der CP/M-Familie, wird ebenfalls mit seinen wichtigsten Befehlen beschrieben. Das Buch enthält Suchhinweise für die unterschiedlichen Anwendungsfälle der Kommandos unter verschiedenen Betriebssystem-Versionen und Tabellen mit deren Fehlermeldungen. Die Kommandos des CP/M-Debuggers DDT bzw. SID runden das Nachschlagewerk ab. Da in der DDR jedoch in größerem Maße das CP/M-ähnliche CPA angewendet wird, dürfte der Kreis der Leser auf die Nutzer von importierten Rechnern beschränkt bleiben.
„TURBO-PASCAL ständig im Griff" wird vom Herausgeber als Referenzhandbuch angepriesen. Referenz heißt Bezugnahme, Hinweis oder Verweis. Es gibt also Hinweise für Computerbenutzer, die mit TURBO-PASCAL arbeiten wollen und PASCAL oder zumindest eine andere Programmiersprache beherrschen.
In den Abschnitten 1 bis 11 wird der Sprachumfang der Version 3.0 im Überblick dargestellt.
Dabei wird auf die Gliederung von Programmen, Grunddatentypen, Vereinbarungs- und Anweisungsteil, Steuerbefehle, Wertzuweisungen und arithmetische Ausdrücke bzw. Funktionen und die Zeichenkettenverarbeitung eingegangen. Die für mich als Datenbankverwalter wichtigen Filebehandlungsroutinen ASSIGN, REWRITE, RESET, SEEK, FILESIZE, FILEPOS und CLOSE und die Verwendung der Datentypen SET und POINTER werden knapp, aber anschaulich beschrieben.
Es werden sowohl Standardprozeduren und -funktionen als auch solche, die vom Benutzer definiert wurden, und spezielle Möglichkeiten der maschinennahen Programmierung erläutert. Im Abschnitt 12 findet man Hinweise zu Grafikanwendung und Tonerzeugung auf dem IBM-PC.
Der Abschnitt 13 „Das Turbo-Pascal-System" erläutert das Hauptmenü und den Aufruf einzelner Komponenten. Er enthält Hinweise für das Arbeiten mit dem Editor, dessen Befehle weitestgehend mit denen von Wordstar übereinstimmen, für das Arbeiten mit dem Compiler sowie die Compilerdirektiven und Fehlermeldungen.
Im Anhang ist eine Zusammenstellung der wichtigsten TURBO-PASCAL-Befehle unter CP/M und MS-DOS enthalten. Ich finde es günstig, daß sich die Hinweise auf die Unterschiede zwischen CP/M und MS-DOS durch alle Abschnitte ziehen.
Warum der Compiler, bis auf Seite 92–94, mit K geschrieben wurde, ist mir nicht klar, wird er doch im TURBO-PASCAL-Hauptmenü mit C geschrieben und auch mit diesem Buchstaben aufgerufen. Die stark gegliederte Darstellung wirkt durch den Einsatz unterschiedlicher Schriftarten für Zwischenüberschriften, Befehle, kurze Beispiele und Erläuterungen sehr übersichtlich.

Dagmar Hemke

Digitale Bildverarbeitung

Von P. Haberaecker, Carl Hanser Verlag München 1985, 377 S., 83 Abb., 29 Übungen und 30 PASCAL-Programme

Das Buch ist besonders für die Einarbeitung in die Problematik der digitalen Bildverarbeitung und der Mustererkennung sowie für das Sammeln von praktischen Erfahrungen auf diesen Gebieten geeignet.
Wesentliche Vorgänge bei der Digitalisierung von Bilddaten werden anschaulich erläutert, wobei der behandelte Stoff logisch und übersichtlich gegliedert ist.
Ausgehend von Schwarz/Weiß-Bildern werden Grauton-, Farb-, Multispektral- und Zeitreihenbilder eingeführt.
Aufbauend auf diesen Definitionen werden folgende Themenkomplexe behandelt:
– Beschreibung digitalisierter Bilddaten durch Maßzahlen
– Speicherung von digitalen Bilddaten
– Datenkompression und -reduktion
– bildliche Reproduktion von digitalisierten gespeicherten Bilddaten
– Bildverbesserung
– Modifikation, Transformation und Anwendung von Operatoren auf digitale Bilder
– Segmentierung von Bildern.
Der letzte Teil des Buches beschäftigt sich mit dem der digitalen Bildverarbeitung verwandten Gebiet der Mustererkennung. Mittels numerischer Methoden der Klassifikation können dabei logisch zusammengehörende Bildinhalte erkannt und gruppiert werden.
Die Anschaulichkeit der Ausführungen wird durch zahlreiche Bilder unterstützt. Jedes Kapitel ist mit umfangreichen Literaturangaben versehen, die besonders relevant für das behandelte Teilgebiet sind.
Die 30 PASCAL-Programme erleichtern den Einstieg in den Problemkreis und ermöglichen sofort das Sammeln praktischer Erfahrungen. Dabei werden neben Programmen zur Berechnung grundlegender Charakteristika, wie Mittelwert, Histogramm, Kovarianzmatrix, auch solche wichtigen Algorithmen wie die 2dimensionale Fouriertransformation, verschiedene Filter, Hauptkomponententransformation und Clusteranalyse bereitgestellt.
Der lehrbuchhafte Charakter des Buches wird durch Übungsaufgaben zu jedem Kapitel unterstützt. Die Auflösung der Aufgaben erfolgt am Ende des Buches.

Dr. R. Gebhardt



---

in FOKUSO 1/88:

Gábor Deák Jahn:
Optische Speicher – Wie werden wir morgen Daten speichern?
(Funktion von CD-Geräten und CD-Speichern, Kodierung zur Verhinderung von Fehlinformationen, weitere optische Speicher, Diagramme zur perspektivischen Entwicklung von optischen Speichern)
Richard Partecke:
Auf dem Wege zum Bio-Computer?
(Ziele der Entwicklung von Computern der 5. Generation, warum Bio-Computer und wie funktionieren diese?)
Jun'ići Sibajama:
Programmierung entsprechend der Logik
(Programmieren mit der Programmiersprache PROLOG nach logischen Gesichtspunkten und Nutzung des Konzepts bei der Entwicklung von Computern der 5. Generation in Japan).
Foto- oder Xerokopien einzelner Artikel können über den Fernleihverkehr bei der Bibliothek der Bergakademie Freiberg bestellt werden.
Übersetzungswünsche können gerichtet werden an:
U. Wilke, Hufelandstr. 31, Berlin, 1055.
Internationale Zeitschrift über Aktuelles in Wissenschaft und Technik, in Esperanto, beziehbar über: Michael Behr, Koburger Str. 83, Markkleeberg, 7113 – Preis: 4 Hefte je 80 Seiten, pro Jahrgang für 24,–M).

W.



Entwicklungen und Tendenzen

Transputer-Karte für den PC

Ein Transputer-Zusatzboard für PCs bringt die Parsytec GmbH, Aachen, auf den Markt. Der Modul mit der Bezeichnung MTM-PC wird in einen Standard-PC-Steckplatz gesteckt. Er verfügt über vier Transputer (T800 oder T414) mit jeweils 1 Megabyte Lokalspeicher zur parallelen Verarbeitung.
Nach Herstellerangaben erreicht ein PC mit dem Transputer-Board bei 20 Megahertz Taktfrequenz und 120 Nanosekunden Speicherzugriff eine Rechenleistung von 40 MIPS (Millionen Operationen pro Sekunde) beziehungsweise 6 MFLOPS (Millionen Gleitkomma-Operationen pro Sekunde).
Zusätzlich zu den vier eingebauten Prozessoren können zwei weitere Transputer-Piggypacks (TRAMs) verschiedener Hersteller für Spezialzwecke aufgesteckt werden.

MP

dBase IV angekündigt

Zur diesjährigen CeBIT stellte die Firma Ashton-Tate die neue Version IV der erfolgreichen dBase-Produktlinie vor. Wesentliche Verbesserungen gegenüber der Vorgängerversion dBase III Plus sind eine deutlich verbesserte Benutzeroberfläche, eine drastisch erhöhte Ausführungsgeschwindigkeit der Programme, die Unterstützung der Datenbank-Abfragesprache SQL und ein automatischer Datenschutz für den Einsatz in Netzwerk-Datenbanken.
Die neue Benutzeroberfläche arbeitet nicht mehr mit dem Assist-Modus, sondern erlaubt über ein Leitmenü in Verbindung mit einer Framework-ähnlichen Benutzeroberfläche auch Nicht-Programmierern die volle Ausnutzung der dBase-Kapazität.
Ein Programmgenerator und ein Compiler sind bereits im System integriert. Die Ausführungsgeschwindigkeit gegenüber dBase III Plus ist je nach Anwendung bis zu neunmal höher.

MP

Computernetz mit 1,4 Milliarden Bit pro Sekunde

Für die Übertragung der in einer Enzyklopädie enthaltenen Informationen benötigt ein in den USA eingeführtes lokales Computernetz weniger als drei Sekunden. Das Unternehmen Scientific Computer Systems Corporation stellte in San Diego das Kabelsystem Vectornet vor, das Daten mit einer Geschwindigkeit von 1,4 Milliarden Bit pro Sekunde übermittelt. Es übertrifft das für die Verbindung von Bürocomputern gebräuchliche Ethernet um das 140fache und das jüngst in Betrieb genommene System Fiber Digital Data Interface für Großcomputer um das 14fache. Schnellere lokale Übertragungsnetze werden nach Einschätzung der Fachleute in den USA vor allem benötigt, weil die Operationsgeschwindigkeit der Computer im zurückliegenden Jahrzehnt ihren Fähigkeiten zur Datenübermittlung weit vorausgeeilt ist. Diese Lücke, die sich in Computerzentren als Hemmschuh erwies, soll geschlossen werden. Als zweiter Grund wird die Notwendigkeit angeführt, grafische Informationen, die durch Computer gewonnen werden, auf Farbbildschirme mit hoher Auflösung zu projizieren. Die bisherigen Übertragungsgeschwindigkeiten zwischen Computer und Bildschirm reichten dazu nicht aus.

ADN

Betriebssystem OS/2 auch von Hewlett Packard

Auch die Firma Hewlett Packard bietet nunmehr für ihre Computer drei Versionen von Microsofts Betriebssystem OS/2 an:
– OS/2 1.0 Version A für HP Vectra ES, ES/12, RS/16 und RS/20
– OS/2 1.0 Version B für die Original HP Vectra-PC sowie HP-spezifisches Zubehör
– OS/2 1.1 mit dem Presentation Manager (ab 1989).
Damit will HP den Weg für leistungsstarke Applikationen ebnen, die für das OS/2 zu erwarten sind. OS/2 benötigt 640 KByte Basisspeicher, 1 MByte Extended Memory (2 bis 4 MByte, wenn mehrere Programme gleichzeitig laufen sollen), eine Festplatte und ein Diskettenlaufwerk mit hoher Kapazität.

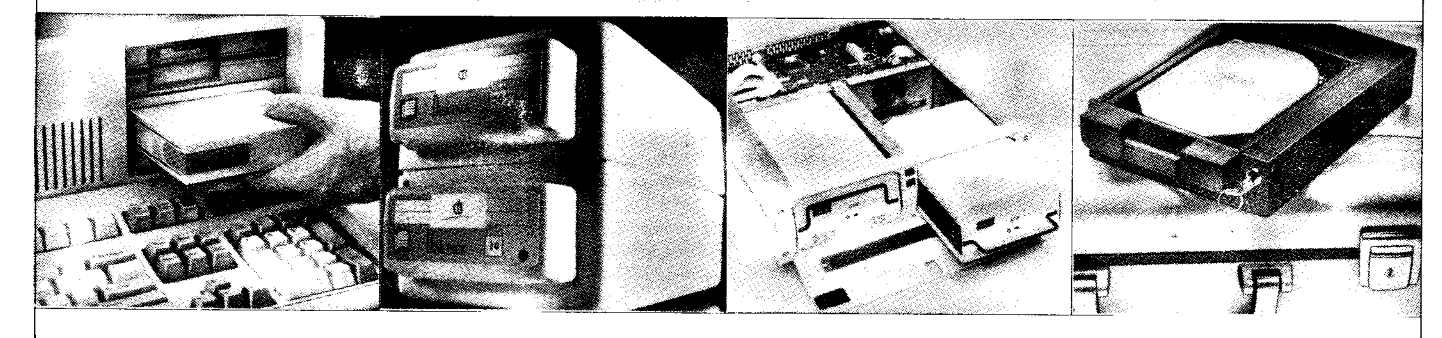
MP

Neurocomputer-Leiterplatte für PC-Serie

Die Firma NEC Corp. arbeitet gegenwärtig an der Entwicklung einer Neuro-Leiterplatte sowie spezifischer „lernfähiger" Software für Desktop-Computervarianten der 16-Bit-Serie 9800, die bis Ende dieses Jahres noch auf den Markt kommen soll.
Diese Neuro-Leiterplatte basiert auf der Grundlage eines Netzes von Schaltkreisen ähnlich den Neuronen im menschlichen Gehirn. Sie arbeitet mit einer Geschwindigkeit von 216 000 Verknüpfungsbefehlen je Sekunde, was der Leistung kostenintensiver Super-Minicomputer entspricht. Es wird eingeschätzt, daß sogenannte Neurocomputer, ausgestattet mit der spezifischen Software, etwa zehnmal so schnell Zeichen und Sprache erkennen und Roboter steuern können, wie herkömmliche Verfahren.

Wi

Mobile Festplatten



In MP 8/88 (4. Umschlagseite) hatten wir über die interessante Lösung von Tandon berichtet, Festplatten als mobile Datenträger zu verwenden. Wie vermutet, haben weitere Hersteller inzwischen die Idee aufgegriffen und ihre Produkte präsentiert. Dazu eine kleine Auswahl:
Plus Development Corp. hatte sich 1985 einen Namen gemacht mit der Hardcard, einer Leiterkarte für PCs, auf der ein 10-MByte-Festplattenlaufwerk (später auf 20 und 40 MByte erweitert) einschließlich Controller untergebracht war. Jetzt stellte Plus das Wechselplattensystem **Passport** vor (Bild 1), ein transportables 20- oder 40-MByte-Laufwerk. In einen normalen Slimline-5,25-Zoll Einbauschacht eines PC, XT, AT oder PS/2-Modells bzw. ein externes Gehäuse wird ein Schubsystem eingebaut, welches einen speziellen Wechselplatten-Modul aufnimmt. Die Kontakte werden dann über Steckverbindungen hergestellt.
Ebenfalls mit 5,25-Zoll-Plattenkassetten arbeitet das Wechselplatten system **SH 20 R** von BiCom. Die handlichen 20-MByte-Kassetten lassen sich unter MS-DOS an allen XT- und AT-kompatiblen PCs nutzen.
Mit dem **Disk-Pack-System** der CTS GmbH steht jetzt ein Wechsel-Festplattensystem nicht nur für Kompatible, sondern auch für den weit verbreiteten Apple Macintosh zur Verfügung. Die Disk-Pack-Box (Bild 2) kann ein oder zwei 20-, 45- oder 70-MByte-Speichermodule aufnehmen; geplant sind Kapazitäten von 100 und 160 MByte. Basis dieser Lösung ist ein doppeltes 3½-Zoll-Winchesterlaufwerk.
Speziell für die Nutzer von Sun-Workstations entwickelte MDB Systems Inc. eine bewegliche Massenspeicherzusatzeinheit als Gehäuse, welches zwei gekapselte Harddisk-Einschübe mit 5,25-Zoll-Laufwerken aufnehmen kann. Die Kapazität (formatiert) liegt zwischen 140 und 630 MByte. Der Einsatz des als **Data Shuttle** (Bild 3) bezeichneten Systems erfordert weder Softwareänderungen noch den Einbau eines Controllers in die Workstation.
Besonders interessant ist die Idee von CTT Computertechnik, für den PC-üblichen 5,25-Zoll-Schacht einen Einbaurahmen zu fertigen (mit den Kontaktleisten an der Rückseite), der eine „Schublade" enthält, in die ein beliebiges 3½-Zoll-Winchesterlaufwerk eingesteckt werden kann. Die **Drive Box** (Bild 4) wird sowohl einzeln als auch komplett mit Laufwerken bekannter Hersteller geliefert.
Den gleichen Gedanken realisiert die Firma Sydec mit dem **Portable-Data-Base-System.** Auch hier wird ein spezieller Rahmen in einem PC-Laufwerksschacht installiert, in den eine vorbereitete 20-MByte-Festplatte leicht austauschbar eingeschoben wird.

MP-We

Mikroprozessor Motorola 88000 mit 100 MIPS

Zwischen den amerikanischen Unternehmen Data General und Motorola soll ein Abkommen über die Entwicklung und Vermarktung einer ultraschnellen Version des Mikroprozessors 88000 unterzeichnet worden sein.
Durch Verwendung von ECL-Technologie und RISC-Architektur sollen ab 1991 für Mainframes, Superminis und Supercomputer Chips mit einer Leistung von 100 MIPS bereitstehen. Zur Zeit ist die CMOS-Version des Motorola 88000 mit 14 bis 17 MIPS Rekordhalter.

Quelle: Computer-Zeitung. – Leinfelde-Echterdingen 19 (1988) 11. – S. 2 Fa

Fortschritte bei Hochvakuum-Schaltkreisen

Nachdem Vakuumbauelemente lange Zeit im Schatten der Halbleiterbauelemente standen, zeichnen sich aufgrund der mit neuen Fertigungsmethoden erzielten Verbesserungen neue vielversprechende Anwendungen ab. Unter Verwendung von Fertigungsprozessen für Halbleiter (Rasterelektronenstrahllithographie) konnten die Vakuumschaltkreise ständig verkleinert werden, so daß es jetzt möglich sein soll, 10000 Bauelemente auf einer Fläche mit einem Durchmesser von 1 mm unterzubringen. Der Vorteil der Hochvakuumbauelemente besteht darin, daß Elektronen sich durch ein Vakuum zehnmal schneller bewegen als die in den gegenwärtig verwendeten Galliumarsenidhalbleitern und zwanzigmal schneller als in Siliziumhalbleitern.
Die Hochvakuumbauelemente sollen die Transistoren nicht aus ihren Anwendungsgebieten verdrängen, können sie aber sinnvoll ergänzen und mit ihnen kombiniert werden. Gegenüber traditionellen Hochvakuumröhren beruhen die neuen Hochvakuumbauelemente auf dem Prinzip der quantenmechanischen Durchtunnelung. Der Elektronenfluß wird dadurch, im Gegensatz zu traditionellen Hochvakuumröhren, nicht durch Wärme erzeugt. Für die sogenannte Kaltdiode kommen Molybdän, Wolfram und Silizium zum Einsatz.
Als Einsatzgebiete der Hochvakuumbauelemente werden Geräte gesehen, die unter extremen Bedingungen arbeiten, z. B. im Weltraum, da sie im Gegensatz zu Halbleiterbauelementen ihre Eigenschaften unter Einwirkung von Strahlen oder extremen Temperaturen nicht ändern, oder Geräte mit extremen Geschwindigkeitsanforderungen. Erstes wichtiges Anwendungsgebiet werden Anzeigeröhren sein, z. B. in Flachbildschirmen für Fernsehapparate und Computer, direkte Satellitenübertragungen mit kleinformatigen Empfangsvorrichtungen, extrem schnelle Supercomputer

Quelle: New York Times vom 18. 5. 1988 Wi

Basistechnologie für Josephson-Computer

Obwohl metallische Halbleiterbauelemente in der jüngsten Vergangenheit durch die sensationellen Meldungen der Hochtemperatursupraleitung von keramischen Materialien in den Hintergrund getreten sind, konnten die japanischen Firmen Fujitsu und NEC mit Erfolgen bei Josephson-Elementen auf metallischer Basis auftreten. So stellte NEC einen 1-Kilobit-Speicherchip vor, der eine Schaltgeschwindigkeit von 570 Pikosekunden bei einer Arbeitstemperatur von 4,2 Kelvin erreichen soll. Der Speicherchip verbraucht eine Leistung von 13 Milliwatt. Fujitsu konnte ebenfalls einen Speicherchip vorstellen, der allerdings nur eine Kapazität von 4 Bit aufweist. Wesentlicher ist der Prozessor, den diese Firma vorstellte. Bei den Experimenten wurden Schaltzeiten von 1,3 Nanosekunden und eine Leistungsaufnahme von 5 Milliwatt erreicht.
Die Erfolge der japanischen Firmen sollen vornehmlich auf die Abkehr von den bisher verwendeten Bleilegierungen und die Hinwendung zu Niobverbindungen zurückzuführen sein. Der praktische Einsatz der Bauelemente wird für Ende der neunziger Jahre erwartet. Bis dahin sind u. a. noch leistungsfähigere und wirtschaftlichere Kühltechniken zu entwickeln.
Hinsichtlich der Konkurrenz der Hochtemperatur-Materialien schätzen die Firmen ein, daß bei den Keramiken noch zu viele Unklarheiten hinsichtlich Theorie und Struktur bestehen und die Stabilität noch nicht ausreicht.

Quelle: Blick durch die Wirtschaft vom 25. 5. 88 Wi

IBM kündigt neue Minicomputerserie an

Die Firma IBM stellte eine neue Minicomputer-Serie mit der Bezeichnung Application System/400 (AS/400) vor, zu der 6 Modelle gehören. Nach Angaben des Unternehmens sollen die neuen Modelle leichter bedienbar sein und dem Kunden die Möglichkeit des Erweiterns bieten, ohne daß die Software geändert werden muß. Gleichzeitig wurden zu der neuen Serie mehr als 1 000 Softwarepakete vorgestellt, die sich besonders für solche Gebiete wie Einzelhandel, Produktion, Bildung, Finanz-, Gesundheits- und Versicherungswesen eignen.
Die AS/400-Serie gilt als Nachfolger der Computerserien System/36 und System/38. Alle Modelle der neuen Serie können mit der gleichen Software arbeiten. Die Programme der älteren Modelle können für die neue Serie kompiliert werden.

Quelle: International Herald Tribune vom 22. 6. 1988 Fa

Forschungsarbeiten für künftige Personalcomputer

Nach Ansicht des Leiters des Hauses der grafischen Datenverarbeitung in Darmstadt, Encarnacao, müssen Personalcomputer in Zukunft alle Spielarten der Musterverarbeitung wie Computergrafik, Animation, Bilderkennung, Sprachsynthese und Spracherkennung realisieren. Der Personalcomputer der Zukunft ist eine multimediale Workstation mit großer Rechenleistung.
Bis 1991 sollen Forschungsarbeiten für einen derartigen Rechner abgeschlossen sein.
An die Workstation werden folgende Forderungen gestellt:
- Der PC reagiert auf zahlreiche Möglichkeiten der Mensch-Maschine-Kommunikation (Tastatur, Maus, Bildschirmgrafik, gesprochene Befehle)
- Einbeziehung großer Speicher, darunter optische Platten großer Kapazität
- Verarbeitung und Speicherung von bewegten Bildern und Sprache
- Ausgabe der Ergebnisse über Drucker, Monitor und Sprachsynthese.

Die künftigen PCs sollen eine sechsfache Leistung gegenüber den gegenwärtigen aufweisen. Die Hälfte der Leistungssteigerung wird über Hardware, die andere Hälfte über effizientere Rechenmethoden erzielt. Bei der Hardware wird auf den Transputer der Firma Inmos zurückgegriffen, es sollen aber auch eigene Chips, besonders für das Erreichen höchster Grafikleistungen, entwickelt werden.

Quelle: Chip. – Würzburg (1988) 6. – S. 30–34 Wi

DEC stellt VAX 8800 und VAX 6200 vor

Die Technik symmetrischen Multiprocessings wurde erstmals in der von der Firma Digital Equipment vorgestellten neuen Generation der VAX-Rechnerfamilie verwirklicht. Die neue VAX 8800-Serie soll dabei eine Leistung erreichen, die um den Faktor 22 über der VAX 11/780 liegt. Die neue Serie, zu der fünf Modelle gehören, ordnet sich trotz der Nutzung neuer technologischer Ergebnisse problemlos in die bestehende VAX-Familie ein. Die Softwarekompatibilität ist gewährleistet.
Durch die Technik des sogenannten symmetrischen Multiprocessing ist es möglich, mehrere Prozessoren durch einen internen Bus miteinander zu betreiben. DEC hat damit 3 unterschiedliche Kopplungsmöglichkeiten für Rechner realisiert. Neben dem symmetrischen Multiprocessing können die Rechner über Netzwerkkabel und -protokolle oder über die Technik des sogenannten VAXclusters (externer Bus, der eine Verbindung zwischen den Prozessoren der Rechner herstellt) miteinander verbunden werden.

Quelle: Computer-Zeitung. – Leinefelden-Echterdingen 19 (1988) 11. – S. 5
Büroreport. – Wien (1988) 3. – S. 14 Wi

Nutzung der Rechentechnik in Forschung und Entwicklung

Nach einer Studie des US-amerikanischen Industrial Research Institute (IRI) werden in den USA im Jahre 1995 90% der in der Industrie tätigen Wissenschaftler und Ingenieure Computer benutzen. Außerdem werden zu diesem Zeitpunkt alle Leiter mit Arbeitsplatzcomputern arbeiten. Im Jahre 1986 nutzten 54,9 Prozent der Fachkräfte und 58,5 Prozent der Leiter Arbeitsplatzcomputer im Dialogbetrieb. Zwischen 1984 und 1986 ist in den Forschungs- und Entwicklungsbereichen von Industrieunternehmen die Arbeit mit lokalen Rechnernetzen von 58,2 auf 92,6 Prozent angewachsen.
Die Hauptanwendungen von Computern sind: Laborautomatisierung, statistische Datenanalyse, Datenbankverwaltung, Modellentwicklung und Simulation. In den nächsten fünf Jahren wird sich die Anwendung in Richtung wissensbasierter Systeme und Expertensysteme entwickeln. Die Leiter nutzen die Rechner zur Budgetplanung, Tabellenkalkulation, Laborplanung, Projektleitung, Textherstellung und zur internen Kommunikation.
In der Intensität der Computerbenutzung liegen der Maschinen- und Gerätebau, die Metallindustrie, die Elektrotechnik/Elektronik und die Chemische Industrie vor den Konsumgüterproduzenten.
Für die Computerhardware werden 16,3 Prozent des Forschungs- und Entwicklungsbudgets ausgegeben. Die Mehrzahl (66,3 Prozent) der Firma gestatten den Anschluß von Heimcomputern an die betriebliche Rechentechnik. Gegenwärtig verfügen 8,6 Prozent des F+E-Personals über einen Anschluß ihrer privat gekauften Rechner an das betriebliche Rechenzentrum.

Quelle: Blick durch die Wirtschaft vom 10. 5. 1988 Wi

IBM entwickelt VHSI-Chip

Einen VHSI-Chip (very high speed integrated) mit einer Strukturbreite von einem halben Mikrometer hat die Firma IBM im Rahmen des vom amerikanischen Verteidigungsministerium finanzierten Forschungsprogramms hergestellt. Damit sei nach Angaben der Firma eine wesentliche Hürde auf dem Weg zur Entwicklung eines 16-Megabit-Speicherchips genommen.

Quelle: Blick durch die Wirtschaft vom 13. 5. 88 Fa

Gate-Arrays mit 1 Million Gattern

Unabhängig voneinander sollen in den Laboratorien von IBM Corp. und TRW Inc. Chips mit einer Gatteranzahl von 1 Million entwickelt worden sein. Den bisherigen Bestwert hielt ein Chip von Motorola mit 500 000 Gattern.
20 000 bis 100 000 Gatter stellen gegenwärtig bei Gate-Arrays in der Produktion den Standard dar.
Ein Chip mit einer 1 Million Gatter könnte z. B. einen ganzen Prozessor einschließlich dazugehöriger Speicher und Hilfslogik für die Außenanschlüsse enthalten. Die Zahl der teuren Leiterplatten könnte wesentlich reduziert werden.
Die Chips von IBM und TRW wurden beide in Silizium-Technologie unter Verwendung der Submikrometer-Struktur hergestellt.

Quelle: Electronics Weekly. – London (1988) 1404. – S. 3 Fa

Laserdiode emittiert im sichtbaren Bereich

Dioden, die im Bereich des sichtbaren Lichtes emittieren, können bei Verwendung zur Datenaufzeichnung auf Optikplatten die Aufzeichnungsdichte um den Faktor 1,5 verbessern. Deshalb wurde von der Firma NEC eine Laserdiode mit einer Wellenlänge von 680 nm entwickelt. Eine weitere Möglichkeit, die Aufzeichnungsdichte zu verbessern, besteht in der Erhöhung der optischen Ausgangsleistung. Die Firma NEC arbeitet gegenwärtig daran, diese von 3,2 mW auf 30 mW zu erhöhen.

Quelle: Elektronik. – München 37 (1988) 8. – S. 7 Fa

Neue Chips kopieren Funktionsweise menschlicher Neuronen

Neuartige Computerchips, die auf ähnliche Art und Weise wie die 100 Milliarden Neuronen des menschlichen Gehirns große Datenmengen simultan verarbeiten, sind von Wissenschaftlern aus den AT & T Bell Laboratories entwickelt worden. Die Forscher bezeichnen das als einen weiteren Schritt voran bei der Konstruktion von Computer-Systemen, die das menschliche Gehirn nachahmen. Nach Angaben des Konzerns fungieren die experimentellen „associative memory"-Chips auf sehr vereinfachter Ebene als elektronische Neuronen. Die Chips sind sechs mal sieben Millimeter groß und enthalten etwa 75000 Transistorfunktionen und 54 einfache Prozessoren. Diese Prozessoren sind durch ein Netz spezifischer Kopplungselemente miteinander verbunden, die programmiert werden können. Von den neuen Mikrochips erwarten die Wissenschaftler eine zehn- bis 100fache Leistungssteigerung.

ADN

Erster dreidimensionaler Chip

Den ersten sogenannten Sandwich-Chip hat das Hughes Research Laboratory in Phoenix (Arizona) entwickelt und vorgestellt. In diesem Chip sind die elektischen Verbindungen dreidimensional durch mehrere Etagen hindurch möglich, so daß sich kürzere Wege ergeben. Der Chip arbeitet nach Angaben seiner Hersteller etwa achtmal schneller als ein herkömmlicher.
Entscheidend für das Funktionieren eines gewissermaßen aus mehreren übereinander getürmten Chips bestehenden 3D-Chips ist ein Material, das sich bei Hitzeeinwirkung nicht ausdehnt, damit sich die Kontaktpunkte der verschiedenen Ebenen nicht verschieben. Geplant sind Chips, die etwa 300mal schneller sind als die heutigen.

ADN

Technik international

Flash-EEPROMs

Die Programmierung von EPROMs erfolgt durch das Tunneln von Ladungsträgern in die Gateelektrode, die in einer Isolationsschicht (meist SiO_2) vergraben ist. Das kann durch Anlegen einer relativ hohen Spannung an die darüber liegende Gateelektrode erreicht werden. Für das Löschen haben sich zwei Methoden durchgesetzt: die Bestrahlung der Gateelektrode mit ultraviolettem Licht sowie die elektrische Löschung. Ihre beiden Hauptvertreter sind zur Zeit die UV-löschbaren EPROMs und die EEPROMs (electrically erasable, programmable read-only memory = elektrisch löschbarer, programmierbarer Nur-Lese-Speicher). International beträgt der Marktanteil von UV-EPROMs 80% gegenüber 20% von EEPROMs.
Die Löschzeiten für UV-EPROMs liegen meist zwischen 10 und 20 Minuten, für EEPROMs betragen sie nur etwa 1 bis 5 Minuten.
Im letzten Jahr machten aber zunehmend die blitzartig löschbaren Flash-EEPROMs von sich reden. Ihre Löschzeiten liegen zwischen 0,5 und 10 Sekunden. Damit sind sie erheblich schneller löschbar als ihre Vorgänger. Ihr zweiter Vorteil ist, daß die Flash-EEPROM-Zelle wie die des UV-EPROMs nur einen Transistor benötigt (siehe Bild 1) und damit höhere Integrationsgrade als beim EEPROM möglich sind.
Da die Flash-EEPROMs wie die EEPROMs kein Löschfenster benötigen, können sie mit einem Plastgehäuse verkappt werden und sind deshalb preislich mit den UV-EPROMs vergleichbar. Die EEPROMs dagegen haben einen 5- bis 10fach höheren Preis. Weil die Flash-EEPROMs natürlich auch (wie die EEPROMs) die Möglichkeit der Umprogrammierung auf der Leiterplatte (in-circuit) bieten, schließen sie die Lücke zwischen den UV-EPROMs und den EEPROMs.
Als Erfinder der Flash-EEPROMs gilt die Firma Seeq Technology. Ihr erster Schaltkreis war der NMOS-Typ 48128 mit 16 KByte Speicherkapazität. Seine Transistor-Zellengröße betrug noch $35\,\mu m^2$. Inzwischen ist der CMOS-Typ 48F512 mit $20\,\mu m^2$ verfügbar. Eine Verringerung der Zellengröße auf $13{,}5\,\mu m^2$ wird für möglich gehalten. Die ersten 1-MBit-Typen von Seeq und Intel sind angekündigt. Für 1991 wird der 8-MBit-Flash-EEPROM erwartet, und um die Jahrtausendwende kann mit dem 64-MBit-Schaltkreis gerechnet werden.
Intel nannte den Herstellungsprozeß für Flash-EEPROMs ETOX (EPROM Tunnel Oxide). Mit ihm wird ein Gateoxid von nur 10 nm Dicke erzeugt. Andere Hersteller arbeiten aber auch mit Gateoxiddicken von 20 bis 50 nm und liegen damit im Bereich der Dikken von konventionellen EPROMs.
Die Zahl der möglichen Programmierzyklen ist noch sehr differenziert. 100 Programmierzyklen garantieren Intel (geplant: 10000) und Seeq (geplant: 1000), die Speicher von Catalyst gestatten bereits 10000 Programmierungen. Die Dauer des Datenerhaltes wird einheitlich mit 10 Jahren angegeben.
Zusätzlich haben einige Hersteller einen Schreibschutz vorgesehen, zum Beispiel bei Betriebsspannungen kleiner 3,5 V oder bei Störimpulsen (kleiner 20 ns). Durch spezielle Algorithmen für das Programmieren (Quick-pulse programming algorithm) und das Löschen (Quick-erase algorithm) werden Programmier- und Löschzeiten von nur einigen Sekunden erreicht. Intel bietet seine 256-KBit-Schaltkreise in zwei Ausführungen an. Der 27F256 hat ein 28poliges DIL-Gehäuse, arbeitet mit Adreßmultiplex und beinhaltet eine $\overline{WE}$-Doppelfunktion. Der 28F256 dagegen hat ein 32poliges Gehäuse und arbeitet ohne Adreßmultiplex. Er besitzt eine sogenannte Befehlsregister-Architektur. Das Befehlsregister steuert sämtliche Funktionen, die den Speicherinhalt verändern und stellt eine direkte Schnittstelle zur CPU dar. Damit kann der Systementwurf für Mikrorechner vereinfacht werden.
Die genannten Vorteile der Flash-EEPROMS lassen natürlich einen Aufschwung ihrer Entwicklung und Fertigung erwarten. Schätzungen zufolge sollen sich die Marktanteile der drei EPROM-Typen bis 1992 folgendermaßen verändern /1/:

UV-EPROMs 54%
EEPROMs 13%
Flash-EEPROMs 33%

In Tafel 1 sind einige Flash-EEPROMs aufgeführt, die bereits bzw. in Kürze verfügbar sind.

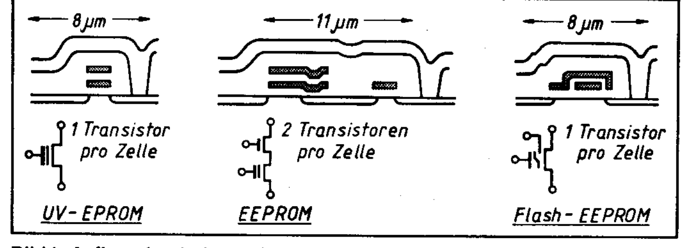


Bild 1 Aufbau der drei verschiedenen EPROM-Zellen

Tafel 1 Flash-EEPROM-Typen

Speichergröße in Bit	Typ	Technologie		Zugriffszeit in ns	Pinzahl	Hersteller
16 K	CAT28C16A	CMOS			24/32	Catalyst
16 K	CAT28C17A	CMOS			28/32	Catalyst
64 K	27F64	CMOS	1,5 um	150/250	28	Intel
128 K	48128	NMOS				Seeq
256 K	CAT28C256	CMOS	1 um			Catalyst
256 K	27F256	CMOS	1,5 um	170/250	28	Intel
256 K	28F256	CMOS	1,5 um	170/250	32	Intel
256 K	TMM28257P	NMOS	2 um	200/250/300	28	Toshiba
256 K	TC58257	CMOS		170/200/250		Toshiba
256 K						NSC
512 K		NMOS		250		Seeq
512 K	48F512	CMOS	1,5 um		32	Seeq
1 M	48F1024	CMOS	1,5 um		32	Seeq

Literatur

/1/ Lineback, J. R.: High-density Flash EEPROMs are about to burst on the memory market. Ele~tronics, Volume 61 (1988) 5, S. 47

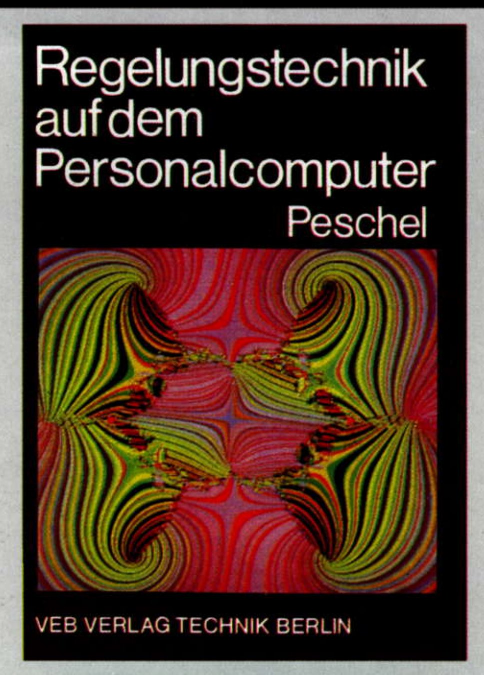
MP-Hk

Ferroelektrische RAMs

Seit kurzem sind neue nichtflüchtige Speicher, die ferroelektrischen RAMs (FRAMs), im Gespräch. Die Entdeckung der Ferroelektrizität geht auf das Jahr 1921 zurück.
Kaliumnatriumtartrat (Seignettesalz) weist ein hohes Maß an elektrischer Polarisierbarkeit auf. Es behält nach Abschalten des elektrischen Feldes eine meßbare elektrische Restfeldstärke. Diese Erscheinung, die u. a. auch bei Blei-Zirkonat-Titanat (PZT), Bariumtitanat und Kaliumhydrogenphosphat (KDP) auftritt, wird beispielsweise für elektroakustische Wandler und Filter, Kondensatoren sowie Lichtmodulatoren (Kerr-Zellen) genutzt.
In zwei US-amerikanischen Firmen (Ramtron und Krysalis) wurden in diesem Jahr die ersten RAMs, die den ferroelektrischen Effekt von PZT nutzen, produziert. Sie haben zwar nur Speichergrößen von weniger als 1 KBit, für 1989 sind aber Kapazitäten von bis zu 256 KBit angekündigt; bis zu 4 MBit werden für möglich gehalten. Durch ihre Zugriffszeit, die im Bereich von Flash-EEPROMs liegt und durch die hohe Anzahl möglicher Schreib-Lese-Zyklen (10^{12}), werden die FRAMs in Zukunft den Flash-EEPROMs (10^5 Zyklen) Konkurrenz machen. FRAMs bieten aber gegenüber EPROMs weitere Vorteile. Wie bei RAMs üblich, liegt die Zeit für einen Programmierzyklus im Bereich der Zeit von einem Lesezyklus. Außerdem können die FRAMs in einer dreidimensionalen Struktur, beispielsweise über einem Prozessorchip, angeordnet werden. Damit kann man eine rasche Erhöhung des Integrationsgrades von FRAMs sowie deren Integration in andere Schaltkreise erwarten.

aus Elektronik 12/1988 *MP-Hk*

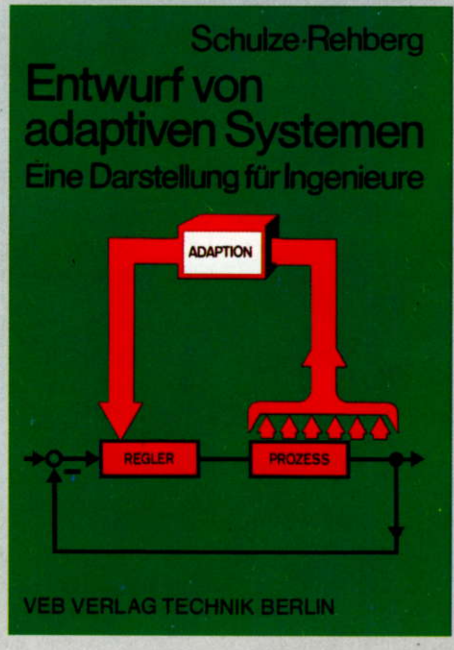
Regelungstechnik auf dem Personalcomputer



Von Prof. Dr. rer. nat. Manfred Peschel. 1. Auflage. 208 Seiten, 41 Bilder, Kunstleder, DDR 22,– M, Ausland 36,– DM.
Bestellangaben: 553 901 8/Peschel, Personalcomputer
Gezeigt wird am Beispiel des Lehrstoffs für das Hochschulfach Regelungstechnik, wie ein Mikrorechner (ausgestattet mit Grafik, Monitor und Disketteneinheit oder Recorder) unmittelbar im Fachunterricht eingesetzt werden kann.
Zu allen wesentlichen Abschnitten des Buches sind lauffähige Programme (40 BASIC-Programme) angegeben, versehen mit geeigneten Testbeispielen und ausgedruckten Grafiken.

Entwurf von adaptiven Systemen

Eine Darstellung für Ingenieure



Von Doz. Dr. sc. techn. Klaus-Peter Schulze und Dr.-Ing. Klaus-Jürgen Rehberg. 1. Auflage. 244 Seiten, 201 Bilder, Leinen, DDR 33,– M, Ausland 42,– DM. Bestellangaben: 553 762 9/Schulze, Entwurf
Aufgrund ihrer bemerkenswerten Eigenschaft, sich selbsttätig an unbekannte oder zeitveränderliche Betriebsbedingungen anzupassen, sind adaptive Systeme in nahezu allen Bereichen der Industrie sehr universell einsetzbar. Die Mikrorechentechnik gestattet auch erstmals die umfassende technische Realisierung. In der Monographie wird sehr anschaulich durch zahlreiche Bilder, Ablaufpläne und Beispiele eine geschlossene Darstellung der wichtigsten Entwurfsprinzipien gegeben.



Grundlagen der problemorientierten Programmentwicklung

Von Dr. sc. techn. Ulrich Lindner und Dr.-Ing. Rainer Trautloft. 2., unveränderte Auflage. 408 Seiten, 153 Bilder, 74 Tafeln, Leinen, DDR 30,– M, Ausland 36,– DM. Bestellangaben: 553 748 5/Lindner, Programmentwickl.



Praxis der Softwareentwicklung

Von Dr.-Ing. Günter Rothhardt. 2., unveränderte Auflage. 296 Seiten, 68 Bilder, 23 Tafeln, Leinen, DDR 30,– M, Ausland 45,– DM. Bestellangaben: 553 751 4/Rothhardt, Software

Auslieferung in diesen Tagen durch den Fachbuchhandel

ISSN 0232-2892
Mikroprozessortechnik, Berlin 2 (1988) 11
S. 321–352

Der VT 32 ist ein 16-Bit-Mikrorechner moderner Architektur mit 32-Bit-Systembus der ungarischen Videoton-Werke. Das Rechnersystem ist insbesondere gekennzeichnet durch
– einen modularen Aufbau, der die Zusammenstellung von verschiedenen, miteinander kompatiblen Rechnern ermöglicht. Das heißt, es können sowohl selbständige Rechner mit einem oder mehreren Arbeitsplätzen als auch Arbeitsstationen, die Teile eines rechentechnischen Systems sind, oder Serverstationen gebildet werden.
– eine Erweiterbarkeit, die eine Anpassung an den jeweiligen Aufgabenumfang ermöglicht.



Videoton VT 32 vorgestellt

Technischer Aufbau

Es wird zwischen drei verschiedenen Modulen unterschieden: Prozessormodul, Speichermodul und Anschlußmodulen. Die Module sind maximal voneinander unabhängig. Die meisten von ihnen können gleichzeitig mehrere Funktionen zur gleichen Zeit ausführen. Über den Systembus kommunizieren sie nach dem Unterordnungsprinzip (asynchron) miteinander.
Als Systembus wurde der leistungsfähige VMEbus ausgewählt.
4 Prioritätsebenen ermöglichen die konkurrierende Nutzung des Busses durch die Module. Gepufferte Peripheriegeräte bzw. Geräte mit veränderlichen Geschwindigkeiten sind über Anschlüsse mit niedriger Priorität gekoppelt. Über den Systembus können Daten mit einer Breite von 8,16 oder 32 Bit übertragen werden, wobei sich die Datenbreite dynamisch, buszyklenweise ändern kann. Der Adreßbereich des Busses beträgt 16 MByte.
Der Prozessormodul hat eine Busverzweigung, den sogenannten lokalen Bus, mit 16 Daten- und 24 Adreßkanälen. Über diesen Bus kann der Datenverkehr parallel zum Datenverkehr des Systembusses erfolgen. Der lokale Bus und das Prozessorinterface beinhalten auch die Möglichkeit des Anschlusses eines Cache-Speichers.

● Prozessormodul

Der Prozessormodul hat folgende funktionelle Teile:
– Zentrale Verarbeitungseinheit: externes Interface, mit 16 Bit, interne Organisation mit 32 Bit, 16 MByte direkt erreichbarer Adreßbereich, 14 Adreßarten, 56 Befehlstypen mit zahlreichen Varianten, 5 Datentypen (Bit, Byte, Wort, Doppelwort, BCD) sowie ein memory mapped-Ein- und -Ausgang
– Einheit für die Speicherorganisation: Sie ermöglicht dem Prozessor die Adressentransformation, den Schutz der Speichersegmente und die Trennung der Befehls- und Datenfelder. Dabei werden Parameter von 32 Speichersegmenten unterstützt. Die Größe des Segments ist variierbar zwischen 256 Byte und 16 MByte.
– interner Speicher (EPROM/RAM) für die Lade-, Test-, Systemstart-, und Diagnoseprogramme
– programmierbarer Zähler/Uhr mit 24 Bit
– zwei sequentielle Interfaces (CCITT V.24) für den Terminalanschluß, für den Anschluß von Druckern mit sequentiellen Interfaces bzw. für den Hostanschluß. Die Übertragungsgeschwindigkeit ist programmierbar im Bereich von 50–19 200 Bit/s.
– programmierbares Parallelinterface. In der Grundausstattung ist ein Centronics-Interface zum Drucker realisiert.
– Systembusinterface und -Steuerung zur Gewährleistung eines Standard-VME-Busses
– internes lokales Businterface sowie Steuerung.

● Speichermodul

Das System enthält mit Paritätsschutz versehene dynamische Speichermodule. Die Kapazität eines Moduls beträgt 512 KByte. Der Adreßbereich des Systems ist 16 MByte. Durch die derzeitige Konfiguration (Anzahl der Steckkartenplätze, Kapazität der Stromversorgung) ist aber die physische Speicherkapazität eingeschränkt auf 2 MByte, d. h. 4 Module.
Die interne Dateiorganisation der Speichermodule erfolgt mit 32 Bit, wobei über den Systembus Dateien mit einer Breite von 8, 16 oder 32 Bit übertragen werden können. Das bedeutet, daß Anschlüsse mit hoher Geschwindigkeit bei direktem Speicherzugriff die 32 Bit des Busses ausnutzen können, womit sich die Belastung des Busses bedeutend verringert. Die Lesezeit des Speichers beträgt 400 ns, die Schreibzeit 370 ns. Die Schaltkreise des Speichermoduls können auch über einen Akku versorgt werden.

● Extern-Speicheranschlußkarte

Über das Standardperipherieinterface ermöglicht sie den Anschluß von zwei Disketteneinheiten (SA 460/SA 850) und vier Festplatten (ST 506/ST 412).

● Kommunikationsmodul

Der Kommunikationsmodul unterstützt vier unabhängige Kanäle. Sie können wahlweise mittels Asynchron-, Bytesynchron-(BSC) bzw. Bitsynchron-(SDLC/HDLS) – Protokollen genutzt werden.
Die Übertragungsgeschwindigkeit ist auf den einzelnen Kanälen programmierbar.

● Grafisches Untersystem

Das grafische Untersystem wird durch ein an den VMEbus angeschlossenes Modulpaar (Steuermodul und Speichermodul) realisiert. Die Hauptaufgabe dieses Untersystems ist die schnelle Ausführung der zweidimensionalen grafischen Operationen und die Unterstützung der peripheren Geräte.
Für das grafische Untersystem wird ein 19-Zoll-Farbbildschirm mit einer Auflösung von 1024 × 768 Punkten verwendet.
Die Anzahl der darstellbaren Farben hängt von der Anzahl der grafischen Module ab:
1 Speicher: 16 Farben aus 4096
2 Speicher: 256 Farben aus 4096
3 Speicher: 4096 Farben.
Einige charakteristische Leistungsparameter:
– Zeichengeschwindigkeit von Vektoren 1 MPixel/s
– Auffüllen von Bereichen und logische Rasteroperationen 256 MBit/s
– Zeichnen von Zeichen 3000 Zeichen/s

● Geräteinterface

Das Geräteinterface beinhaltet die für die Steuerung des parallelen IEEE-488(GPIB)-Busses notwendigen technischen Mittel.

● Lokalnetzanschluß

Er ermöglicht eine Kopplung mehrerer VT 32 über verhältnismäßig kleine Entfernungen (ca. 1 km) im lokalen Netz. Der physische Anschluß und der Datenaustausch sind entsprechend der Ethernet-Spezifikation realisiert.

● Modulweiterentwicklungen

Das Modulangebot des Systems wird ständig weiterentwickelt. Von den gegenwärtig in Entwicklung befindlichen Modulen ist die an den Systembus anschließbare Steuereinheit erwähnenswert (grafischer Prozessor und Bit-Map-Speicher), die für die Steuerung von Schwarz-weiß- und farbigen Bildschirmen mit unterschiedlicher Auflösung und grafischen Eingabegeräten geeignet ist.

● Periphere Geräte

Zur Grundkonfiguration eines VT 32 gehören
– 2 bis 6 alphanumerische oder grafische Terminals mit mittlerer Auflösung
– Minidisketteneinheit (5 1/4") mit 720 KByte und 5 1/4"-Festplatteneinheit mit 20–40 KByte (in der Grundausrüstung können maximal 2 Disketten- und 2 Festplatteneinheiten untergebracht sein).
Matrixdrucker (120 Zeichen/s oder Kettendrucker (300, 600 oder 900 Zeilen/min)

Software

Betriebssystem
Der Rechner VT 32 läuft mit dem Mehrnutzerbetriebssystem DMOS, das kompatibel zu UNIX ist.

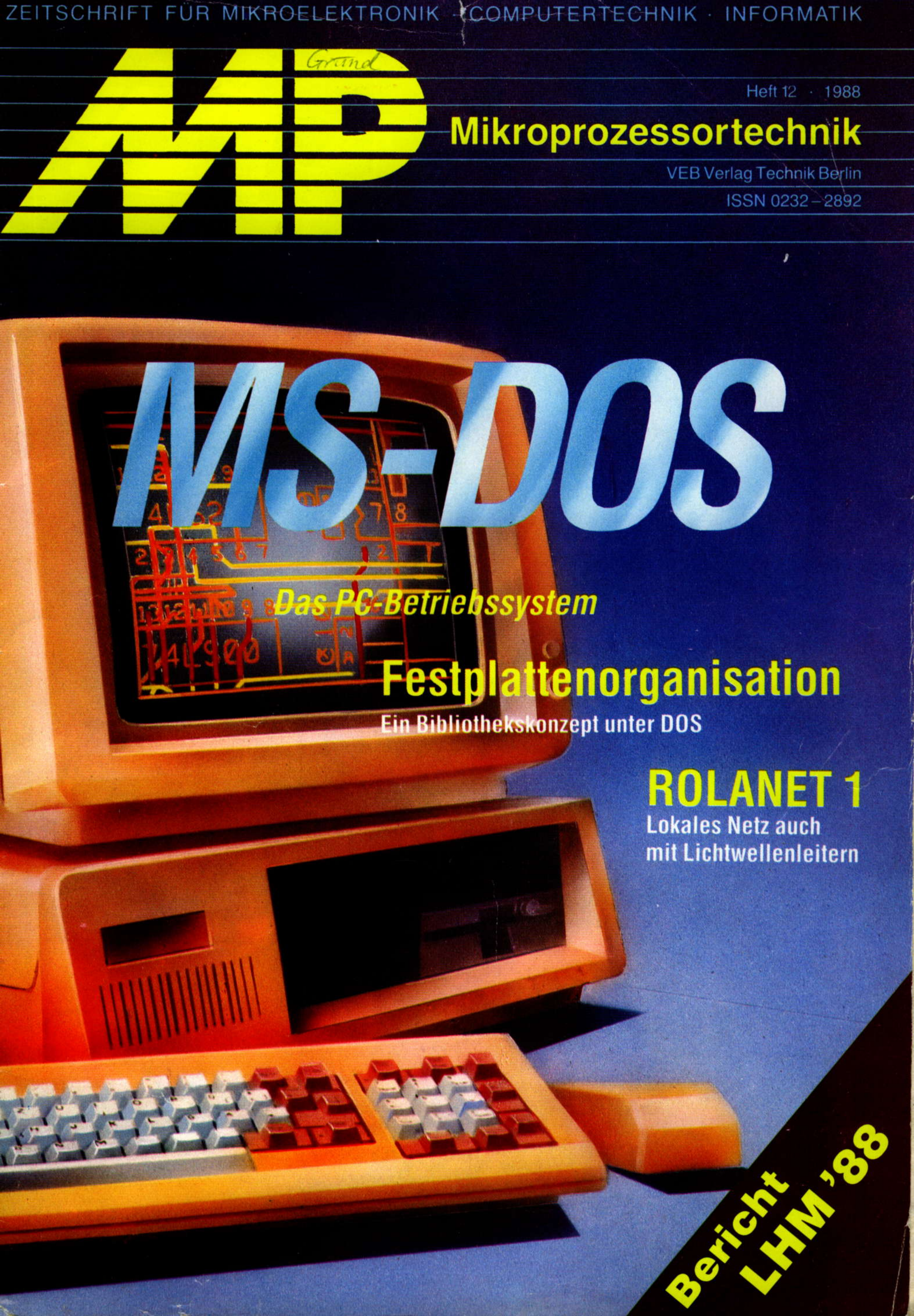
Sprachen
Sowohl die Mehrzahl der Systemprogramme als auch das Betriebssystem sind in C geschrieben. Außer C sind noch die Compiler für Assembler, FORTRAN 77, COBOL level II und PASCAL (ISO) vorhanden.

Kommunikationssoftware
Unterstützung von:
– Asynchronverbindungen mit:
• uucp (UNIX to Unix copy) zwischen zwei VT 32
• Kermit: zwischen VT 32 und R 11
zwischen VT 32 und PC
– Synchronverbindungen mit:
• Emulator IBM 2780 (BSC)
• Emulator IBM 3274 (BSC, SDLC)
– Lokalnetzlösungen:
• EXLOC 4.1 (VT 32 – VT 32)
• EXLOC 4.2 (VT 32 – VT 11)

Bildschirmorientierter Editor
Der Editor stellt in erster Linie ein Mittel für die Programmänderung dar, ist aber auch durch die Ausnutzung der Möglichkeiten der Bildschirmbehandlung für einfachere Aufgaben der Textaufbereitung geeignet.

Pixel Manager – PixMan
PixMan (Pixel Manager) ist ein im EPROM des grafischen Untersystems abgespeichertes Programmpaket. Interfaces mit höherem Niveau (z. B. CGi, GKS) bauen auf den Diensten des PixMans auf. Das Programmpaket beinhaltet über 200 Funktionen.

J. Ilse



ZEITSCHRIFT FÜR MIKROELEKTRONIK · COMPUTERTECHNIK · INFORMATIK
MP
Heft 12 · 1988
Mikroprozessortechnik
VEB Verlag Technik Berlin
ISSN 0232-2892
MS-DOS
Das PC-Betriebssystem
Festplattenorganisation
Ein Bibliothekskonzept unter DOS
ROLANET 1
Lokales Netz auch
mit Lichtwellenleitern
Bericht
LHM '88

❶ *Lichtsatz-Serie 2000 von Linotype*

❷ *MCS Integrator Serie 19 von compugraphic*

❸ *INPUT-Texterfassung mit EC 1834 am DOSY von elaplan buchholz/HELL*

❹ *Satzsystem TOPSET von elaplan buchholz/HELL*

❺ *Bildungscomputer als Steuerrechner für Schülerexperimente*

❻ *Projektionsplatte DATASHOW von Kodak*

❼ *LC-Display data vision von Geha*

❽ *Videoprinter P 70 B von Mitsubishi*

❾ *Pocket Computer PC-1600 von Sharp*

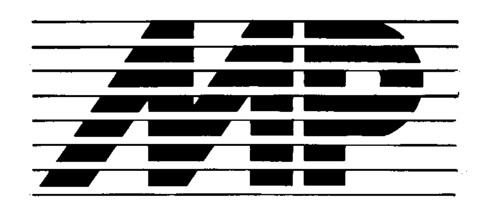
❿ *LCD-Tastatur The Board von Hohe Electronics*



Bilder von der Leipziger Herbstmesse 1988

Lesen Sie dazu unseren Bericht in diesem Heft

Zeitschrift für Mikroelektronik · Computertechnik · Informatik
2. Jahrgang · ISSN 0232–2892



Mikroprozessortechnik, Heft 12 · 88

Herausgeber Kammer der Technik, Fachverband Elektrotechnik

Verlag VEB Verlag Technik, Oranienburger Str. 13/14, DDR – 1020 Berlin; Telegrammadresse: Technikverlag Berlin; Telefon: 28700, Telex: 011 2228 techn dd

Verlagsdirektor Klaus Hieronimus

Redaktion Hans Weiß, Verantwortlicher Redakteur (Tel. 2870371); Herbert Hemke, Redakteur (Tel. 2870203); Sekretariat Tel. 2870381

Gestaltung Christina Bauer

Titel Herbert Hemke, Tatjana Stephanowitz

Lizenz-Nr. 1710 des Presseamtes beim Vorsitzenden des Ministerrates der Deutschen Demokratischen Republik

Gesamtherstellung Druckerei Märkische Volksstimme Potsdam

Redaktionsschluß: 19. Oktober 1988

AN (EDV) 49837

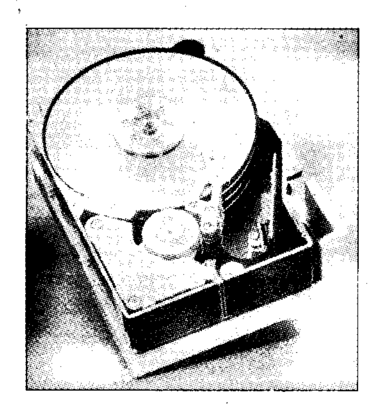
Erscheinungsweise monatlich 1 Heft

Heftpreis 5,– M, Abonnementspreis vierteljährlich 15,– M; Auslandspreise sind den Zeitschriftenkatalogen des Außenhandelsbetriebes BUCHEXPORT zu entnehmen.

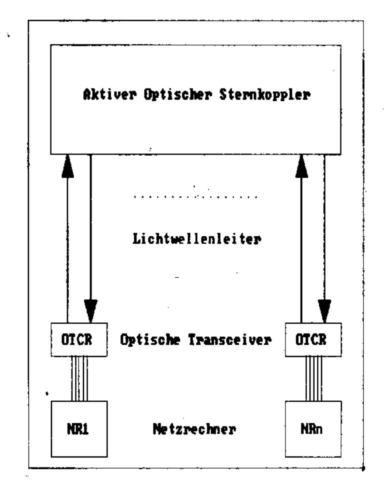
Bezugsmöglichkeiten
DDR: sämtliche Postämter; *SVR Albanien:* Direktorije Quendrore e Perhapjes dhe Propaganditit te Librit Rruga Konference e Pezes, Tirana; *VR Bulgarien:* Direkzia R.E.P., 11a, Rue Paris, Sofia; *VR China:* China National Publications Import and Export Corporation, West Europe Department, P.O. Box 88, Beijing; *ČSSR:* PNS – Ustřední Expedicia a Dovoz Tisku Praha, Slezská 11, 120 00 Praha 2, PNS, Ústredna Expedicia a Dovoz Tlače, Pošta 022, 885 47 Bratislava; *SFR Jugoslawien:* Jugoslovenska Knjiga, Terazija 27, Beograd; Izdavačko Knjižarsko Produzeće MLADOST, Ilica 30, Zagreb; *Koreanische DVR:* CHULPANMUL Korea Publications Export & Import Corporation, Pyongyang; *Republik Kuba:* Empresa de Comercio Exterior de Publicaciones, O'Reilly No. 407, Ciudad Habana; *VR Polen:* C.K.P.i.W. Ruch, Towarowa 28, 00-958 Warszawa; *SR Rumänien:* D.E.P. Bucureşti, Piaţa Scînteii, Bucureşti; *UdSSR:* Sämtliche Abteilungen von Sojuzpečat' oder Postämter und Postkontore; *Ungarische VR:* P.K.H.I., Külföldi Elöfizetési Osztály, P.O. Box 16, 1426 Budapest; *SR Vietnam:* XUNHASABA, 32, Hai Ba Trung, Hà Nội; *BRD und Berlin (West):* ESKABE Kommissions-Grossobuchhandlung, Postfach 36, 8222 Ruhpolding/Obb.; Helios-Literatur-Vertriebs-GmbH, Eichborndamm 141–167, Berlin (West) 52; Kunst und Wissen Erich Bieber OHG, Postfach 46, 7000 Stuttgart 1; Gebrüder Petermann, BUCH + ZEITUNG INTERNATIONAL, Kurfürstenstraße 111, Berlin (West) 30; *Österreich:* Helios-Literatur-Vertriebs-GmbH & Co. KG, Industriestraße B 13, 2345 Brunn am Gebirge; *Schweiz:* Verlagsauslieferung Wissenschaft der Freihofer AG, Weinbergstr. 109, 8033 Zürich; *Alle anderen Länder:* örtlicher Fachbuchhandel; BUCHEXPORT Volkseigener Außenhandelsbetrieb der Deutschen Demokratischen Republik, Postfach 160, DDR – 7010 Leipzig und Leipzig Book Service, Talstraße 29, DDR – 7010 Leipzig

Zum Titelbild

Das Disk Operating System von Microsoft, das Betriebssystem des IBM-PC und seiner Kompatiblen, stellen wir Ihnen auf der **Seite 355** näher vor. Hierzu haben wir auf der 3. und 4. Umschlagseite die Befehle des MS-DOS in einer Referenzkarte zusammengestellt (für Version 3.3 vollständig und für Version 4.0 im Überblick).



Im Beitrag auf der Seite **Seite 359** wird Ihnen ein Ordnungskonzept für Festplatten im DOS vorgestellt, das auf einer Aufteilung der Festplatte in wahlweise ein, zwei oder mehr logische Laufwerke und auf den Laufwerken in sogenannte Bibliotheken beruht. Dieses Ordnungskonzept soll helfen, ständig neu hinzuzufügende Programme sinnvoll in die vorhandene Struktur einzubauen.



Der Beitrag „ROLANET 1 mit Lichtwellenleitern" auf der **Seite 360** stellt ausgehend von den elektrischen Schnittstellen die optischen Komponenten Transceiver, Sternkoppler, Repeater und Multitransceiver dieses lokalen Netzes vor. Die aufgezeigten Netztopologien belegen die Ausdehnung der Anwendungsbreite von ROLANET 1 bei weiterer Nutzung der bisherigen Hard- und Softwarekomponenten.

Inhalt

Vorschau

Für Heft 1/1989 haben wir für Sie unter anderem Beiträge zu folgenden Themen vorbereitet:

- Künstliche Intelligenz
- Manufacturing Automation Protocol (MAP)
- Transputer
- Digitale Signalprozessoren

Entwicklung neuer Mikroprozessoren in Erfurt

In den zurückliegenden zehn Jahren wurde das im VEB Mikroelektronik „Karl Marx" Erfurt produzierte Bauelementesortiment von 22 auf 188 Typen erweitert.
Zum 40. Jahrestag der DDR im Oktober 1989 sollen ein Jahr vorfristig die Serienproduktion eines schnellen 16-Bit-Mikroprozessors aufgenommen und erste funktionstüchtige 32-Bit-Mikroprozessoren zur Verfügung gestellt werden. Diese volkswirtschaftlich bedeutsamen Verpflichtungen übernahmen Mitte Oktober die Genossen der SED-Grundorganisation des Forschungszentrums im Erfurter Stammbetrieb des Kombinates Mikroelektronik auf ihrer Berichtswahlversammlung.

aus ND vom 13. Oktober 1988

Produktionsanlage für keramische Chipgehäuse

Eine neue Fertigungsstätte zur Herstellung von hermetischen Mehrschicht-Keramikgehäusen für Chips wurde Anfang Oktober im Stammbetrieb des Kombinats Keramische Werke Hermsdorf übergeben. Sie ermöglicht die volle Bedarfsdeckung bei diesen für das Mikroelektronikprogramm der DDR wichtigen Bauteilen aus neuartiger technischer Keramik. Noch bis Jahresende sollen 700000 Schaltkreisgehäuse hergestellt werden.
Der Generaldirektor des Hermsdorfer Kombinates, Manfred Schneider, würdigte bei der Inbetriebnahme die Einsatzbereitschaft der Bau- und Ausrüstungskollektive, die das Investitionsobjekt in zweieinhalbjähriger Bauzeit errichteten. Nahezu ein Drittel der Ausrüstungen ist vom kombinatseigenen Rationalisierungsmittelbau entwickelt und hergestellt worden.

ADN

Erste Bildungscomputer von Robotron übergeben

Die ersten 30 Bildungscomputer sind Anfang Oktober vom VEB Robotron-Meßelektronik „Otto Schön" Dresden übergeben worden. Sie werden in der Volksbildung, im Hoch- und Fachschulwesen und der Berufsbildung erprobt. Der A5105 ist der erste Rechner, der eigens für den Unterricht in der Abiturstufe, in polytechnischen Zentren, in Hoch- und Fachschulen sowie Berufsschulen entwikkelt wurde. Er bietet gegenüber den anderen DDR-Kleincomputertypen die bislang umfassendsten Anschlußmöglichkeiten für Drucker, Schreibmaschinen, Magnetbandkassetten und Fernseher sowie für spezielle Unterrichtsmittel und Labortechnik (s. auch MP 10/88).

ADN

Neue Lösungen und Trends bei der Software-entwicklung

Fast die Hälfte aller Ingenieurtätigkeiten in entwickelten Industriestaaten wird derzeit rechnergestützt durchgeführt. Dieser Anteil nimmt weiter zu. Die Rolle des Menschen als Beherrscher der Technik gewinnt daher immer mehr an Bedeutung. Das erklärten Prof. Dr. Gustav Olling (USA) und Prof. Dr. Detlef Kochan (DDR) als Teilnehmer der Mitte des Jahres veranstalteten internationalen Konferenz „PROLAMAT 88". Nach ihren Worten arbeiten Wissenschaftler, Techniker und Ingenieure in Industriestaaten derzeit sehr intensiv an der Software zur flexiblen Produktionsautomatisierung.
Auf neue Lösungen und Trends verwiesen auf der Konferenz, die von der Internationalen Föderation für Informationsprozesse IFIP einberufen worden war, unter anderen Fachleute aus Japan, der DDR, aus der Sowjetunion und den USA. So stellte Hiroshi Katoh von der Nissan-motor-company (Japan) ein komplexes CAD/CAM-System vor, das bereits in diesem Unternehmen angewendet wird. Durch die Kopplung von mehr als 1000 CAD/CAM-Arbeitsstationen mit einem Großrechner erhält damit jeder zweite Ingenieur der Company ständigen Zugang zu betrieblicher Software.
Die Dresdner Konferenz habe die Aufmerksamkeit darauf gelenkt, von gegenwärtig vorhandenen Insellösungen der automatisierten Fertigungsvorbereitung und Produktion nunmehr zu völlig geschlossenen Systemen überzugehen, sagte Prof. Olling. Am Ende dieses Systems müsse es seiner Meinung nach eine automatische Rückkopplung in Form von Erfahrungen, die sich in der Software niederschlagen, geben. Über erste Erfolge auf diesem Gebiet berichteten Wissenschaftler aus den USA.
Anerkennend über das Niveau der computerintegrierten Fertigung (CIM) in der DDR äußerten sich ausländische Konferenzteilnehmer beim Besuch Dresdner Industriebetriebe. Prof. Gustav Olling, der in der Chrysler Motors Corporation tätig ist, nannte als eine der eindrucksvollsten Lösungen „Made in GDR" ein CAD/CAM-System der Technischen Universität Dresden für die Bearbeitung doppelt gekrümmter Flächen.
Gute derartige Lösungen, beispielsweise für die Bearbeitung von Karosserieteilen, habe es bisher nur für rotationssymmetrische und prismatische Teile, die ein einfacheres geometrisches Verhalten aufweisen, gegeben, schätzte er ein. Übereinstimmend machten Prof. Dr. Olling und Prof. Dr. Kochan darauf aufmerksam, die Fachleute dürften ihre Kräfte nicht nur auf die Entwicklung von Technologien konzentrieren. Expertensysteme könnten nur so gut sein wie der Mensch. Demzufolge müsse man stärker die Rolle des Menschen bei der computerintegrierten Fertigung erforschen, denn „der beste Großrechner sitzt zwischen den Ohren des Menschen".

ADN

Wissenschaftskooperation im Elektronikbereich

Rund 40 Prozent der neuen Erzeugnisse aus dem Industriebereich Elektrotechnik und Elektronik werden unter Mitwirkung von Forschungseinrichtungenrealisiert. Auf dem Gebiet der Mikroelektronik und Rechentechnik ist der Prozentsatz doppelt so hoch. Um in kürzester Zeit neue Erzeugnisse und Verfahren bereitzustellen, müssen alle verfügbaren Potenzen der industriellen Forschung und Entwicklung sowie der wissenschaftlichen Einrichtungen ausgeschöpft werden. Darauf verwies Felix Meier, Minister für Elektrotechnik und Elektronik, in einem Pressebeitrag. Wachsende Bedeutung erlange die enge und auf vertraglicher Basis gestaltete Kooperation zwischen Kombinaten, Betrieben und wissenschaftlichen Institutionen der Akademie der Wissenschaften der DDR sowie Universitäten und Hochschulen.
Zwischen den 15 Kombinaten des Ministeriumsbereichs und Forschungseinrichtungen wurden im vergangenen Jahr 58 Koordinierungs- und 875 Leistungsverträge abgeschlossen.
Mit der Technischen Universität Dresden, die Partner von 13 Kombinaten des Industriezweiges ist, besteht das Ziel, bis 1990 Spitzenleistungen unter anderem bei der Verringerung der Strukturbreiten hochintegrierter mikroelektronischer Schaltkreise und der Entwicklung schneller Mikroprozessorsysteme zu erreichen.
Ein hervorstechendes Merkmal der Wissenschaftskooperation, so der Minister, sei der Ausbau der materiell-technischen Basis an den Forschungseinrichtungen mit Unterstützung der Kombinate. Dazu zählen der Aufbau eines Informatikzentrums des Hochschulwesens mit Hilfe des Kombinates Robotron und eines gemeinsamen Zentrums für Elektromaschinenbautechnologien der TU Dresden und des Kombinates Elektromaschinenbau.

ADN

Dialog

Farbfernseher am Kleincomputer

Seit kurzem bin ich im Besitz eines KC 85/3 und freue mich auf jeden Beitrag über den KC 85/3 in der MP.
In dem Artikel „Bild- und Tonanschluß mit RGB-Qualität für Fernsehgeräte der Serie 4000" in der MP 2/88, S. 61, hat sich ein Fehler eingeschlichen: Der Transistor VT 7601 ist kein SC 307, sondern ein SC 236 laut Schaltplan.
Da ich einen Farbfernseher des Typs RC 6075 von Robotron habe, möchte ich ebenfalls einen Computeranschluß einbauen. Der RC 6075 hat jedoch ein anderes SY-Modul (HG-Modul 022.737); deshalb habe ich Probleme, das FBAS-Signal richtig anzuschließen.
Wo muß ich das FBAS-Signal anschließen, und welche Bauelemente muß ich noch zusätzlich einfügen? *Reiner Wilhelm, Kirchmöser*

Wer kann unserem Leser helfen?

Hohe Erneuerungsraten bei Computern und Druckern

Die Jahresproduktion leistungsfähiger Drucker wird sich bis 1990 im Büromaschinenwerk „Ernst Thälmann" Sömmerda auf mehr als 150000 Geräte erhöhen. Hohe Steigerungsraten streben die Kollektive des größten Robotron-Betriebes auch bei Personalcomputern und Konsumgütern an. Voraussetzung dafür sei eine hohe Erneuerungsrate der Produktion bei gleichzeitiger Anwendung effektiver Fertigungstechnologien, betonte Betriebsdirektor Dr. Dieter Jordan auf einer im September durchgeführten Intensivierungskonferenz. Durchschnittlich 40 Prozent aller Erzeugnisse verlassen das Werk jährlich als Neu- oder Weiterentwicklung. Dazu gehören 1988 der 16-Bit-Computer EC 1834 sowie ein Laserdrucker.
Dr. Jordan verwies darauf, daß mit der Überleitung neuer Geräte zugleich auch der Produktionsaufwand gesenkt werden müsse. Mit einer neuen Generation schneller Nadeldrucker soll sich der Arbeitszeitaufwand um rund 30 Prozent im Vergleich zur gegenwärtig produzierten Baureihe verringern. An Schwerpunktaufgaben der Rationalisierung arbeitet der Betrieb mit Instituten der Akademie der Wissenschaften, Universitäten und Hochschulen zusammen. Er ist Alleinhersteller für Drucker und produziert 80 Prozent der Personalcomputer in der DDR.

ADN

EG setzt sich gegen protektionistisches USA-Handelsgesetz zur Wehr

Die in der EG zusammengeschlossenen westeuropäischen Staaten sind nicht gewillt, das vor wenigen Wochen vom Kongreß in Washington verabschiedete eindeutig protektionistische neue Handelsgesetz hinzunehmen (s. auch MP 11/88). Sein rechtliches Instrumentarium ermöglicht es der USA-Regierung, nach Gutdünken über Handelspraktiken anderer Länder oder ausländischer Firmen zu befinden und sie mit Sanktionen wie Strafzöllen oder Importbegrenzungen zu belegen, wenn, nach Auffassung der USA, Interessen der Vereinigten Staaten beeinträchtigt werden. Das kann selbst Unternehmen treffen, die in Drittländern gegen USA-Handelsbestimmungen verstoßen.
Eigennutz und Willkür, so schätzt man bei der EG ein, sind so Tür und Tor geöffnet. Entsprechend scharf haben die Außenminister auf ihrer Ratstagung Mitte September in Brüssel gewarnt, dieses Gesetz könne zu Handlungen und Verhältnissen führen, die den Regeln des internationalen Handels widersprechen, wie sie beispielsweise im Rahmen des internationalen Zoll- und Handelsabkommens (GATT) einst auch von den USA anerkannt wurden.

ADN

Das Betriebssystem MS-DOS

Prof. Dr. Winfried Kalfa
Technische Universität Dresden,
Informatikzentrum des Hochschulwesens

Einführung

Die Firma Microsoft hatte von der Firma Seattle Computer Products die Lizenz für das Betriebssystem QDOS (Quick and Dirthy Operating System) erworben. Noch stark unter dem Einfluß von CP/M (Ein geübter CP/M-Nutzer findet sich sofort in MS-DOS zurecht!) wurde dieses Betriebssystem unter dem Namen MS-DOS (Microsoft Disk Operating System) als Version 1.0 1981 auf den Markt gebracht. Die große Verbreitung fand das **MS-DOS** unter dem Namen PC-DOS, als IBM alle Rechte erwarb und es zum Standardbetriebssystem für seine Arbeitsplatzrechner PC-Junior, PC/XT und PC/AT deklarierte. Mit schätzungsweise mehr als 15 Millionen Installationen (1987) ist es das am weitesten verbreitete Betriebssystem. Der Version 1 folgten die Versionen 2.xx mit einem völlig neuen Filessystem und einer Reihe von neuen bzw. erweiterten Kommandos und Funktionen. Mit dem IBM PC/AT kamen die Versionen 3.xx, die gekennzeichnet sind durch

- mehrere Diskettenformate
- länderspezifische Modifikationen (Alphabet, Tastatur)
- neue externe Kommandos
- Konfigurationsmöglichkeiten
- neue bzw. erweiterte Dienste, insbesondere für in Netze eingebundene PCs.

Auf der Basis der Versionen 3.xx existieren einige zusätzliche Optionen, wie

- **MS-WINDOW:** für Fenstertechnik
- **MS-NET:** für Vernetzung von PCs
- **MS-GDI:** für grafische Verarbeitung.

Die letzte Version von MS-DOS bzw. PC-DOS ist 4.0 (siehe auch MP 10/1988, Seite 317). Das Nachfolgesystem ist das Betriebssystem OS/2, das für das neue Personal System/2 mit dem 32-Bit-Mikroprozessor i 80386 entwickelt wurde. Kompatibel zu MS-DOS ist das Betriebssystem DCP (Disk Control Program) vom VEB Kombinat Robotron, wobei die Versionsnummern übereinstimmen.

MS-DOS ist ein Betriebssystem für Personalcomputer (PC) und wie CP/M ein Ein-Nutzer- und Ein-Prozeß-System. Es wird Wert auf definierte Schnittstellen innerhalb des MS-DOS gelegt. Der Einfluß von UNIX auf die höheren Versionen ist unverkennbar. Dazu zählen das Filessystem ab V. 2.0, das Umleiten von Standard-Ein-/Ausgaben, das Einrichten von Pipelines und Filtern in der COMMAND-Schicht und COMMAND selbst.

Letztlich dient das MS-DOS als Basis für alle kommerziellen Anwendungen von Arbeitsplatzrechnern. Es stehen nicht nur Programmierumgebungen für alle wesentlichen Sprachen zur Verfügung, sondern auch Applikationssysteme, wie Wordstar, WORD, dBASE, Kalkulationsprogramme usw. Das MS-DOS ist nicht für Steueraufgaben geeignet und vorgesehen. Es gibt keine Prozeßverwaltung, wenngleich erste Ansätze in Form von Systemdiensten für Programmladen, -beenden usw. existieren.

Die Anforderungen an das MS-DOS ergeben sich aus den hauptsächlichen Einsatzgebieten

- CAD-Anwendungen
- Softwareentwicklung
- Betriebswirtschaft
- wissenschaftlich-technische Aufgaben
- Büroautomatisierung.

Die zunehmende Verbreitung von MS-DOS führte zu neuen Versionen mit erhöhter Leistungsfähigkeit und zu Zusatzoptionen, die durch das flexible Konzept nachladbarer Treiber sowie ergänzender Systemfunktionen leicht in MS-DOS einzufügen sind. Dazu zählen die bereits oben genannten Optionen, aber auch Multitaskergänzungen. Entsprechend dem Entwurfsziel von MS-DOS als Einnutzersystem für PC und den Randbedingungen der Hardware (i 8086-Schaltkreisfamilie, externe Speicher) ergaben sich die folgenden Leistungsmerkmale:

- Mit MS-DOS kann nur ein Nutzer über ein Terminal (Konsole) arbeiten (*Personal*computer).
- Es gibt keine Prozeßverwaltung, so daß Multiprogramming und Echtzeitverarbeitung nicht unterstützt werden.
- Das MS-DOS ermöglicht nutzerfreundliche „schlüsselfertige" Systeme. Nach dem Einschalten meldet sich automatisch eine Anwendung.
- Es existiert ein einfaches, sicheres und den gängigen Anforderungen entsprechendes Filesystem mit folgenden wesentlichen Aspekten:
  - beliebige Anzahl von Files
  - strukturierte Fileverzeichnisse
  - sequentieller Zugriff mit variabler Satzlänge
  - beliebiges Positionieren im File
  - Zugriffsschutz
  - Datum und Uhrzeit.
- Die Speicherung von Files ist auf Disketten mit unterschiedlichen Formaten und auf Festplatten (Winchesterlaufwerken) möglich.
- Es existiert eine klare Schichtung des Betriebssystems mit guten Möglichkeiten der Konfigurierung.
- Neben der Dialogverarbeitung ist auch Stapelverarbeitung (Batch Processing) möglich.
- Die konsistente Schnittstelle der Dienste für Geräte und Files erlaubt ein beliebiges Umleiten von Ein-/Ausgaben.
- Programme können unter COMMAND durch Pipelining gekettet werden.
- Ab der Version 3.10 gibt es Dienste zur Unterstützung der Nutzung verkoppelter MS-DOS-Rechner.
- MS-DOS ist ein offenes System, das heißt, es können für neue Geräte eigene Driver in das System eingebunden werden, und es können eigene Optionen und Dienste als Ergänzungen des Betriebssystems eingefügt werden.

Schichtenstruktur

Das Betriebssystem MS-DOS ist wie andere moderne Betriebssysteme modular aufgebaut. Diese Modularität drückt sich vor allem in einer Bereitstellung von Diensten aus. Unter einem Dienst wollen wir die Ausführung einer Operation über einem oder mehreren Operanden verstehen. Wenn der Operand ein Registerinhalt ist, kann dieser zum Beispiel durch eine Operation inkrementiert werden. Wenn der Operand ein File (Datei) ist, gibt es Operationen zum Eröffnen und Schließen, zum Erzeugen und Löschen usw. Es steht also der funktionelle Aspekt im Vordergrund, weshalb an Stelle von Dienst auch oft Funktion als Synonym verwandt wird.

Der Dienst wird von dem jeweiligen Modul an einem Dienstzugriffspunkt bereitgestellt. Die Beziehungen der Module sind durch Schnittstellen (Interfaces) gekennzeichnet mit den folgenden Merkmalen

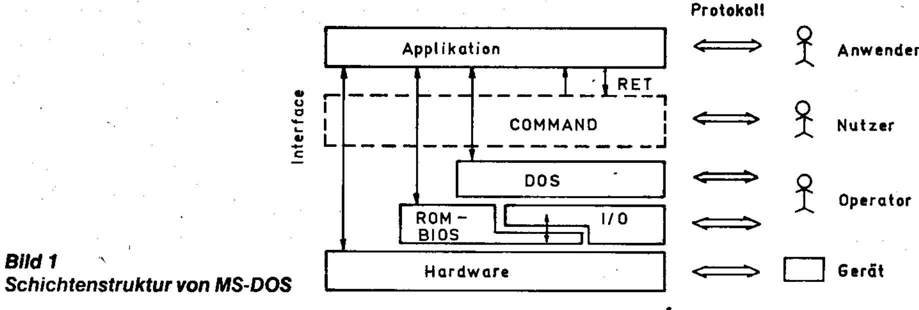


Bild 1
Schichtenstruktur von MS-DOS

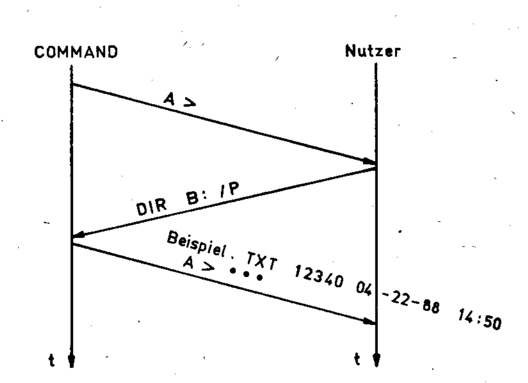


Bild 2 Beispiel für Protokollablauf zwischen COMMAND und Nutzer

– Operanden
– Operationen (Dienste, Funktionen)
– Form des Dienstzugriffes.

Darüber hinaus ist es sinnvoll, mehrere Module zu Schichten zusammenzufassen, wobei es nur Schnittstellen zwischen Schichten gibt (Bild 1).

Einige dieser Module realisieren aber auch eine Kommunikation des Rechnersystems mit der Umwelt, die an Hand von Protokollen abgewickelt wird. Der Nutzer verständigt sich mit dem Betriebssystem über ein Protokoll ebenso wie der Anwender mit einem Applikationsprogramm. Auf dem Bildschirm erscheint durch das COMMAND die Ausgabe des Prompts, worauf der Nutzer mit einem Kommando antwortet. Es folgen die Kommandoausgaben und wieder das Prompt vom COMMAND (Bild 2).

Im MS-DOS kann man die folgenden Schichten unterscheiden:
– Applikation bzw. **COMMAND**
– Disk Operating System (**DOS**)
– Input-/Outputsystem (**I/O**)
– Read Only Memory Basic Input Output System (**ROM-BIOS**)
– Hardware.

Jede dieser Schichten bietet Dienste an einer Schnittstelle, die von Microsoft fest vorgegeben ist. Diese exakte Schnittstellenbeschreibung ist einer der wesentlichen Gründe für die enorme Verbreitung von MS-DOS.

Neben diesen Schnittstellen sind noch einige Systemdaten und die Aufteilung des Hauptspeichers definiert (Tafel 1). Wie die Module des Betriebssystems implementiert werden, ist dann gleichgültig, solange die Schnittstellen eingehalten werden.

Tafel 1 Speicheraufteilung von MS-DOS

Adresse	Bereich	Größe
FFFFF		1024K
	ROM-BIOS	
FE000		1016K
	ROM-BASIC	
F6000		
	ROM-NUTZER	
F4000		
	reserviert	
F0000		960K
	ROM/RAM Speichererweiterungen	
C0000		768K
	reserviert	
BC000		
	Farbgrafik-Bildschirmspeicher	
B8000		736K
	reserviert	
B1000		
	Monochrom-Bildschirmspeicher	
B0000		704K
	reserviert	
A0000		640K
	COMMAND transient	
	System-Stack	
	Nutzerbereich	
08000		32K
	COMMAND resident	
	Driver (zusätzliche)	
	DOS-Puffer	
	DOS (MSDOS.SYS)	
	BIOS (IO.SYS)	
00600		
	DOS-Daten	
00500		
	BIOS-Daten	
00400		
	Interruptvektoren	
00000		

Bis auf die nur intern im Betriebssystem nutzbare Schnittstelle des I/O sind alle Schnittstellen durch den Applikationsprozeß nutzbar. Die genaue Darstellung der Dienste ist die eigentliche Beschreibung des Betriebssystems für den Programmierer.

Für die Hardware wird ein Gerätesystem vorausgesetzt, das auf der Schaltkreisfamilie i 8086/88 (K 1810 WM86) mit einem entsprechenden Bus beruht. Damit ist eine relative Adressierung des Hauptspeichers bis 1 MByte und eine absolute Adressierung von Ports bis 64 KByte für den Peripherieanschluß verbunden.

Das *ROM-BIOS* befindet sich in einem besonderen Nur-Lese-Speicher (ROM) auf der Systemplatine. Seine Adressen liegen mit FE000 ... FFFFF im Adreßraum des normalen Hauptspeichers. Neben Programmen für den Anlauf des Rechnersystems sind im ROM Routinen des Betriebssystems MS-DOS für die Bedienung der Geräte enthalten. Damit brauchen sich Programmautoren nicht mehr um die teilweise sehr komplizierte Programmierung der Geräte zu kümmern. Es stehen Dienste für die im MS-DOS erlaubten Geräte zur Verfügung. Diese Dienste beziehen sich auf idealisierte, sogenannte virtuelle Geräte. Es ist uninteressant, ob es ein Typenraddrucker SD 1152 oder ein Nadeldrucker K 6314 oder ob der Drucker über einen USART, eine PIO eines Nebenbusses oder über IFSS angeschlossen ist. Wichtig für das virtuelle Gerät Drucker ist die Ausgabe eines Zeichens an den Drucker, wobei es ein druckbares Zeichen oder ein Steuerzeichen sein kann. Der Aufruf der ROM-BIOS-Dienste erfolgt über sogenannte Traps

Tafel 2 Virtuelle Geräte der BIOS-Schnittstelle

Interrupt-nummer	Virtuelles Gerät	Dienst
5	CON	Drucken Bildschirminhalt
8	CON	Bildschirm-Ein-/Ausgabe
10	CLOCK	Zeitgeber
11		Ausrüstungstest
12		Hauptspeichergröße
13		Disketten-Ein-/Ausgabe
14	AUX; COM_n	asynchrone Datenübertragung
15		Kassetten-Ein-/Ausgabe
16	CON	Tastatur-Ein-/Ausgabe
17	LPT, PRN_n	Drucker-Ein-/Ausgabe
19		Systemstarten
1A		Tageszeit und Datum

(synchrone Softwareinterrupts) des Prozessors i 8086/88. In Tafel 2 sind die Dienste zusammengestellt. Die **BIOS-Schnittstelle** wird durch folgende Aspekte charakterisiert:
– Dienste für virtuelle Geräte
– Schnittstelle als Softwareinterrupt
– Parameterübergabe über Register bzw. über Datenbereiche des Hauptspeichers, deren Adressen in Registern übergeben werden.

Das *I/O* liefert keine Schnittstelle, die von Applikationsprozessen genutzt werden kann. Es dient im wesentlichen zur Korrektur und eventuellen Erweiterung des ROM-BIOS. Da das I/O beim Start des Betriebssystems von der Diskette geladen wird, kann es im Gegensatz zum ROM-BIOS leicht geändert werden. Das I/O ist im PC-DOS das File IBMBIO.COM, im MS-DOS und im DCP 1700 3.20 das File IO.SYS und im DCP 3.2 das File BIO.COM. Es hat drei Aufgaben zu erfüllen:
– Anpassung von DOS an das ROM-BIOS
– Korrektur von ROM-BIOS-Fehlern
– Einbindung von Drivern für neue Geräte.

Da alle Dienste über Softwareinterrupt aufgerufen werden, braucht das I/O nur den Interruptvektor in der Interruptsäule zu ändern, die ROM-BIOS-Implementation durch eine I/O-spezifische zu ersetzen.

Im Mittelpunkt des MS-DOS steht die Schnittstelle des DOS. Es wird beim Start des Systems aus dem File IBMDOS.COM bzw. MSDOS.SYS (für MS-DOS) bzw. DOS.COM (für DCP) in den Hauptspeicher geladen. MSDOS.SYS und IO.SYS sind sogenannte *versteckte System*-Files, die durch ein übliches DIR-Kommando nicht angezeigt werden. Sie müssen auch als erstes File auf der Systemdiskette bzw. im MS-DOS-Festplattenbereich stehen. Die **DOS-Schnittstelle** hat die folgenden wesentlichen Merkmale:
– Schnittstelle als Softwareinterrupt (im wesentlichen über INT21)
– einheitliche Schnittstelle für Dienste an Geräten und Files
– Verwaltung der Betriebsmittel Hauptspeicher, Zeit und Programme
– Parameter bzw. Adressen von Parametern werden über Register vermittelt.

Die bisherigen Schichten haben kein ausgeprägtes Protokoll zur Verständigung mit der Umwelt, wenn man von einigen Besonderheiten, wie Fehlermeldungen und Auslösen besonderer Aktionen durch Erkennen bestimmter Tastencodes im Tastaturdriver, absieht. Die eigentliche Kommunikation mit der Umwelt erfolgt durch das COMMAND (File COMMAND.COM) über ein Protokoll gemäß Bild 2. Das COMMAND liefert aber auch eine Schnittstelle nach oben in der Weise, daß ein File fn.BAT mit Kommandos interpretiert werden kann. Die Ausführung dieser Kommandos erfolgt aber in derselben Schicht, in der auch das COMMAND liegt. Wenn der Hauptspeicherplatz für ein Kommando (Jedes ausführbare Programm ist auch ein Kommando!) nicht ausreicht, wird gegebenenfalls der transiente Teil von COMMAND überspeichert, nämlich der Kommandointerpreter. Existiert dann keine Systemdiskette bzw. Festplatte mit dem File COMMAND.COM, kann MS-DOS nicht weiterarbeiten.

Das **Protokoll COMMAND-Umwelt** ist gekennzeichnet durch
• interaktive Ausführung von internen Kommandos
– zur Filearbeit
– für die Bildschirmsteuerung
– für Datum und Uhrzeit
– zur Änderung der Standard-Ein-/Ausgabe
• Umleitung von Ein-/Ausgaben der Standardperipherie
• ein sogenanntes Pipelining zur Weiterleitung der Standardausgaben eines Kommandos als Eingabe für ein zweites
• interaktive Ausführung beliebiger externer Kommandos.

Alle Möglichkeiten des Protokolls sind auch an der *Schnittstelle* des *COMMAND*, das heißt des Stapelverarbeitungsteils (Batchprozessor), möglich. Kommandos sind die Dienste dieser Schnittstelle. Darüber hinaus existieren
• Dienste zur Ablaufsteuerung mit: Zyklus; Bedingung; unbedingtem Sprung
• aktuelle Parameter beim Aufruf eines fn.BAT-Files zur Übergabe an die formalen Parameter der fn.BAT-Files.

MS-DOS erlaubt jedoch auch die Einbindung einer eigenen Kommunikationskomponente in Anlehnung an das Betriebssystem UNIX.

Geräte

In MS-DOS werden standardmäßig eine Reihe von Geräten behandelt:
- Magnetplatten
- Tastatur
- Zeitgeber
- Drucker
- serielle und parallele Anschlüsse.

Für diese Geräte gibt es Dienste des MS-DOS, aber auch Kommandos, die sich auf Geräte beziehen, zum Beispiel DATE und TIME.

Filesystem

Das Filesystem ist das Kernstück des MS-DOS. Es gibt im MS-DOS zwei unabhängige Filesysteme:
- das satzorientierte Filesystem (CP/M-ähnlich, Dienste über FCB ansprechbar)
- das zeichenorientierte Filesystem (UNIX-ähnlich, Dienste über Handle ansprechbar).

Für beide ist charakteristisch, daß sie nicht systemweit sind, sondern daß für jedes Laufwerk ein selbständiges Filesystem existiert. Für beide Filesysteme ist die Inanspruchnahme der Dienste durch die Klammerung:

```
OPEN file
CLOSE file
```

zwingend vorgeschrieben.

Das satzorientierte Filesystem liefert Dienste nur für Sätze, auf die sequentiell oder direkt durch Angabe einer Satznummer zugegriffen werden kann. Die Satzlänge ist dabei variabel bei der Eröffnung angebbar. Die Verwaltung dieser Files erfolgt nur im einzigen Wurzelverzeichnis (Rootdirectory) jedes Laufwerkes. Damit ist die Anzahl der verwalteten Files durch die Größe des Rootdirectory festgelegt. Für neue Applikationen sollte dieses Filesystem nicht mehr verwendet werden. Der Zugriff zu den Files erfolgt durch Angabe eines Namens sname, der aus Angabe des Laufwerkes d, des Filennamens fn und des Filetyps (Extension) ft besteht. Die explizite Angabe von sname lautet in der COMMAND-Schicht:

```
  sname :: = [d:] fname
      d :: = A:B:C ...
  fname :: = fn[.ft]
     fn :: = zeichen | fn zeichen
     ft :: = zeichen | ft zeichen
zeichen :: = A|...|Z|a|...|z|0|...|9|!|$|%|&
             |'|(|)|@|^|_|{|}|~
```

Zu beachten ist dabei, daß fn maximal 8 und ft maximal 3 Zeichen sind und daß COMMAND grundsätzlich eine Umwandlung in Großbuchstaben vornimmt. ft hat für DOS keinerlei Bedeutung, verschiedene Kommandos fordern jedoch einen bestimmten Filetyp. Als Zeichen sind auch sogenannte Platzhalter (Wildcards) erlaubt:

∗ ersetzt 1 oder mehrere Zeichen, wobei ∗ für alle *restlichen* Zeichen von fn und ft steht,
? ersetzt *ein* Zeichen.

Für einen Platzhalter steht ein beliebiges erlaubtes Zeichen, so daß alle dazu passenden Files angesprochen werden. **d:** kann entfallen, wenn sich auf das aktuelle Laufwerk bezogen wird.

Die explizite Angabe von sname in der DOS-Schicht besteht immer aus genau 12 Zeichen

```
ddfnft
dd:: = 0|1|2...
```

dd=0 steht für das sogenannte aktuelle Laufwerk, dd=1 für A usw. Fehlende Zeichen in fn und ft sind mit Leerzeichen (Blanks, 20H) aufzufüllen. Wildcards sind dort erlaubt, wo es sinnvoll ist.

Das aktuelle Laufwerk kann durch ein Kommando **d:** oder durch einen DOS-Dienst eingestellt werden.

Das zeichenorientierte Filesystem liefert Dienste für Zeichenfolgen; eine Satzstruktur wird nicht realisiert. Das System (DOS) verwaltet einen aktuellen Zeiger in der Zeichenfolge. Die DOS-Dienste beziehen sich immer auf den aktuellen Stand des Zeigers. Dieser Zeiger kann beliebig durch einen DOS-Dienst verschoben werden. Das Einfügen von Zeichen ist nicht möglich! Der Zugriff zu einem File erfolgt nicht durch Angabe von sname, sondern durch Angabe eines Filezugriffsnamens zname, bestehend aus Laufwerk LW, Pfadname pfad (path) und Filename fname. Die Filenamen sind in einer Hierarchie von Directories gespeichert (Bild 3). Die einzelnen Directories werden wieder durch das Filesystem selbst verwaltet. Damit kann man eine unbegrenzte Anzahl von Files im MS-DOS benutzen.

```
zname :: = lw pfad fname
 pfad :: = [\] dir [\ dir...]\
  dir :: = fn
```

Die Trennung der einzelnen Directorynamen im Pfadnamen erfolgt durch einen sogenannten Backslash \. Der 1. Backslash bedeutet dabei das Rootdirectory. Durch ein Kommando oder durch einen DOS-Dienst kann man ein *aktuelles* Directory einstellen. Will man von einem aktuellen Directory ein File erreichen, braucht man den vorangehenden Teil des Filezugriffsnamens nicht anzugeben. Der Unterschied der Angabe in der COMMAND- und der DOS-Schicht besteht lediglich darin, daß in der DOS-Schicht der Filezugriffsname mit 00H abzuschließen ist. Oft sind Kommandos (ausführbare Programme) und Files in verschiedenen Directories. Deshalb gibt es in der COMMAND-Schicht einen sogenannten Suchpfad, der sich mit einem Kommando oder einem DOS-Dienst einstellen läßt. Über den Suchpfad kann sich COMMAND auf Files des Typs fn.EXE, fn.COM und fn.BAT beziehen. Falls COMMAND ein Kommando zu suchen hat, sucht er diesen in der Reihenfolge:
- aktuelles Directory
- Rootdirectory
- Directory gemäß 1. Suchpfad
- Directory gemäß 2. Suchpfad usw.

Programme und Prozesse

In dem Ein-Prozeß-System MS-DOS ist eine Prozeßverwaltung an sich überflüssig. Es könnten jedoch der Applikationsprozeß und der Prozeß für den aufgerufenen Betriebssystemdienst parallel arbeiten. Das ist aber im MS-DOS nicht vorgesehen. Betriebssystemdienste werden prozedural genutzt, das heißt, der Applikationsprozeß muß warten, bis der Dienst vollständig erbracht wurde.

Bei der Verkopplung von MS-DOS-Rechnern treten dann natürlich Probleme auf. Zusatzoptionen regeln die notwendige Parallelarbeit von Applikationsprozeß und Kommunikationssoftware. Ab der Version 3.00 werden von MS-DOS lediglich Funktionen
- zum Zugriffsschutz von Files von mehreren Rechnern
- zum wechselseitigen Ausschluß kritischer Abschnitte
- zum entfernten Gerätezugriff

bereitgestellt.

Für jede durch das Betriebssystem im Hauptspeicher angelegte Prozedur gibt es ein PSP (Programm Segment Prefix), in dem Informationen über diese Prozedur abgelegt sind. Weitere Informationen sind in der Prozedurumgebung (Environment) enthalten. Der PSP wird durch MS-DOS vor dem Programmtext im Hauptspeicher angelegt.

Das eigentlich Neue in MS-DOS besteht darin, daß
- das Laden und Starten von Prozeduren als Betriebssystemdienste bereitstehen
- mehrere selbständige Prozeduren gleichzeitig in einer Kette existieren können
- Programmtexte resident im Hauptspeicher angelegt werden können.

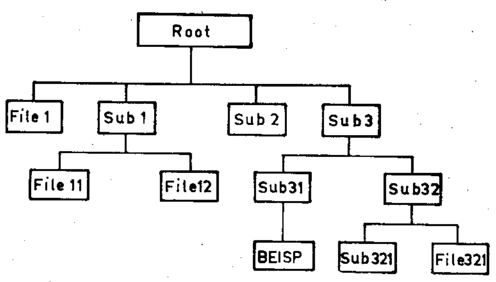
Die Prozessorzuteilung an die Prozeduren wird nicht durch das Betriebssystem, sondern durch die Prozeduren selbst erledigt (Bild 4). Prozeduren können dabei über drei Adressen verlassen werden, nämlich über
- Rückkehr zur Elternprozedur
- Unterbrechungsprozedur
- Fehlerprozedur.

Ausführbare Programme, die den Programmcode für Prozeduren enthalten, werden wie Files verwaltet. Es werden zwei Arten unterschieden: **fn.COM** und **fn.EXE**.

COM-Files sind dadurch gekennzeichnet, daß der Programmcode maximal 64 KByte umfaßt. Alle Befehle dieses Programmcodes sind durch einmaliges Laden des Codesegmentregisters CS erreichbar. Das Laden dieses Programmcodes erfolgt sehr schnell. **.EXE**-Files können beliebig lang sein. Das Codesegmentregister CS muß dynamisch



Bild 3
Filehierarchie



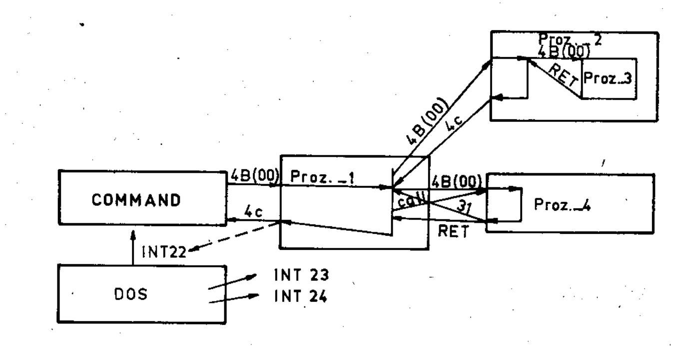


Bild 4
Prozedurverwaltung im MS-DOS

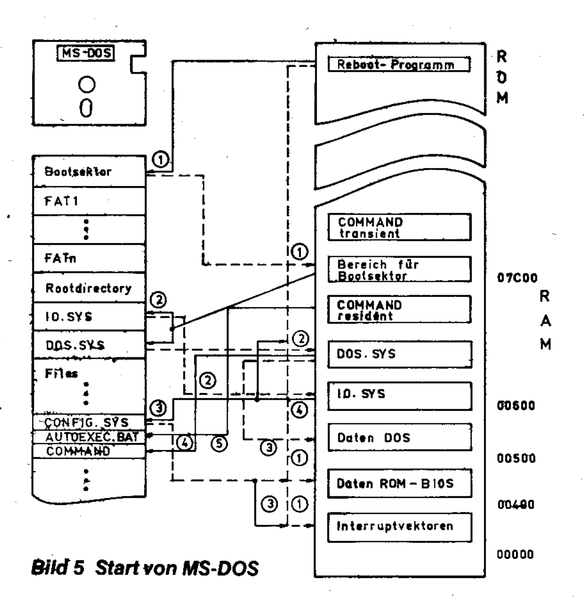


Bild 5 **Start von MS-DOS**

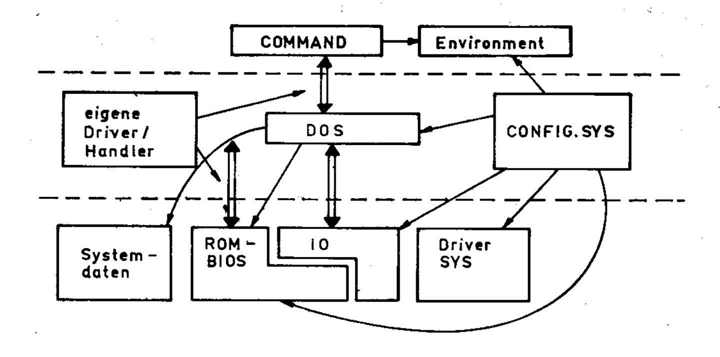


Bild 6 ***Möglichkeiten der Konfigurierung des MS-DOS***

mehrfach umgeladen werden. Dazu stehen in einem Kopf des Files Verschiebeinformationen, die beim Laden an die entsprechenden Stellen in den Programmcode eingefügt werden.

Betriebsmittel

In MS-DOS werden die Betriebsmittel
- Programmtext
- Hauptspeicher
- Zeit

verwaltet, wobei eine echte Verwaltung nur des Hauptspeichers erfolgt.
Eine eigentliche Programmtext-Verwaltung wird nur für den transienten Teil der COMMAND-Prozedur durchgeführt, der den Kommandointerpreter und die internen Kommandos enthält. Dieser Teil liegt am oberen Ende des Hauptspeichers und wird zuletzt durch COM-Prozeduren überspeichert. Jedesmal, wenn dieser Teil angesprochen werden soll, wird von dessen Speicherbereich eine Prüfsumme gebildet, um zu überprüfen, ob der ursprüngliche Programmtext noch erhalten ist.
Für Nutzerprozeduren findet eine solche Prüfung nicht statt, es können also durchaus mehrere Exemplare desselben Programmtextes in den Hauptspeicher geladen und als Prozedur gestartet werden.
Neben der internen Verwaltung des Hauptspeichers durch das Laden von Programmtexten gibt es die folgenden Dienste für die Hauptspeicherverwaltung über INT21H:
- HS anfordern <AH> = 48H
- HS freigeben <AH> = 49H
- HS-Bereich ändern <AH> = 4AH.

Mit dem Laden von Programmtexten bzw. dem Dienst *Hauptspeicher anfordern* werden Hauptspeicherbereiche definiert, die durch ihre Segmentadresse gekennzeichnet werden. Sie können aufgegeben bzw. in ihrer Länge verändert werden. Die Strategie der Hauptspeichervergabe kann abgefragt bzw. gesetzt werden:

first fit
best fit
last fit.

Da es keine eigentliche Prozeßverwaltung gibt, spielen Datum und Zeit nur die Rolle eines Kalenders bzw. einer Uhr, die gesetzt und abgefragt werden können.

Start des MS-DOS-Rechners

Das Betriebssystem wird in folgenden Schritten durch Netzeinschalten bzw. Betätigen der Tasten CTRL-ALT-DEL, was zum Reboot führt, geladen (Bild 5).

ROM-BIOS

- Hardwaretest, Bestimmung der aktuellen Konfigurationsparameter und deren Ablage im Hauptspeicher 00400H...00493H
- Interruptvektoren einstellen
- Lesen des Boot-Sektors (cyl = 0, head = 0, sec = 1) in den Hauptspeicher 07C00H...07DFFH und Start der Boot-Prozedur bei 07C00H.

Boot-Prozedur

- Laden des Files IO.SYS von Diskette unmittelbar nach dem Directory
- Laden des Files MSDOS.SYS von Diskette unmittelbar nach IO.SYS
- Start von IO.SYS

IO.SYS

- Ersetzen bzw. Parametrisieren der BIOS-Driver
- Ersetzen oder Anfügen von nutzerspezifischen Drivern, falls das File CONFIG.SYS existiert
- Start des MSDOS.SYS.

MSDOS.SYS

- Tabellen im DOS zurücksetzen
- Laden von COMMAND.COM und Starten des Initialisierungsteils.

COMMAND.COM

- Der Initialisierungsteil startet das Batch-File AUTOEXEC.BAT, falls es existiert.
- Bestimmung der Ladeadresse für Nutzerprozeduren (der Initialisierungsteil wird überlagert)
- Start des residenten Teils von COMMAND.

Der residente Teil enthält neben den COMMAND-Daten lediglich
- die Standardroutine für Fehler
- die Standardroutine für Abbruch
- sich selbst als Standardroutine für Prozedurende
- die Routine zum Prüfen der Notwendigkeit des Neuladens des transienten Teils
- die Routine zum Neuladen des transienten Teils.

Konfigurierung des Systems

Die Autoren von MS-DOS haben eine Konfigurierung des Betriebssystems prinzipiell vorgesehen. Damit ist nicht ein freizügiger Austausch von Teilen des MS-DOS gemeint. Dazu müßten alle internen Schnittstellen bekannt sein, das ist aber das bestbehütetste Geheimnis von Betriebssystemherstellern. In MS-DOS wird darunter vielmehr eine Parametrisierung vorhandener Module des Betriebssystems verstanden. Das erfolgt
- beim Start des Systems durch den INIT-Modul von IO.SYS unter Verwendung des Files CONFIG.SYS
- beim Start des Systems durch Abarbeiten des Files AUTOEXEC.BAT
- dynamisch während des Betriebes durch spezielle externe und interne Kommandos bzw. DOS-Dienste
- durch nutzereigene Kommandos, die eigene Driver bzw. Module für alte bzw. neue Betriebssystemdienste im Hauptspeicher resident machen.

Bild 6 zeigt im Überblick, wer wo etwas konfigurieren kann.

Literatur

/1/ Kalfa, W.: DCP. Technische Universität Dresden, Lehrheft, Dresden 1988
/2/ Biethan, G.: MS-DOS/PC-DOS. Kurz und bündig. Würzburg: Vogel-Verlag 1986
/3/ Anwendungsbeschreibung für Hard- und Software für den PC EC 1834. VEB Robotron Buchungsmaschinenwerk Karl-Marx-Stadt 1988
/4/ Anleitung für den Bediener, Betriebssystem. ebenda
/5/ Anleitung für den Assemblerprogrammierer. ebenda
/6/ Anleitung für den Bediener, Hardware. ebenda
/7/ De Voney, C.: Das MS-DOS-Kompendium. München: Markt & Technik 1985
/8/ Duncan, R.: MS-DOS für Fortgeschrittene. Wiesbaden: Vieweg 1987
/9/ MS-DOS 3.1. Programmer's Reference Manual, Microsoft. Haar: Markt & Technik 1986
/10/ Neidhold, T.: Assemblerprogrammierung von 8086/88-Computern. Technische Universität, Lehrheft, Dresden 1988
/11/ Norton, P.: Die verborgenen Möglichkeiten des IBM PC. München: Carl Hanser Verlag 1985
/12/ Smode, D.: Das große MS-DOS-Profi-Arbeitsbuch. München: Franzis-Verlag 1987
/13/ Voelzing, P. P.: MS-DOS im Detail. Vaterstetten: IWT-Verlag 1985
/14/ Kalfa, W.: DCP – ein Betriebssystem für Personalcomputer. Leipzig: Fachbuchverlag 1989

⊠ KONTAKT ☏

Technische Universität Dresden, Informatikzentrum des Hochschulwesens, Mommsenstr. 13, Dresden, 8027; Tel. 4575282

Festplattenorganisation – Ein Bibliothekskonzept für die Harddisk unter DOS

Christian Hanisch, Berlin

Festplatten mit einer Kapazität zwischen 10 bis 50 MByte und mehr dominieren als externe Speichermedien für die 16-Bit-PC und lösen die mit Kapazitäten unterhalb 1 MByte vorhandenen Disketten im Routinebetrieb ab.
Durch die große Kapazität der Festplatte, auf der logisch nicht zusammengehörende Programme und Daten physisch gemeinsam gespeichert werden, treten sehr bald organisatorische Probleme vielfältiger Art auf.

Die Möglichkeit des hierarchisch untergliederten Verzeichnisses in ROOT- und SUB-DIRectory gibt zwar theoretisch eine Vielzahl möglicher Handhaben, Ordnung zu schaffen, doch die psychologisch-arbeitsmethodischen Aspekte des jeweiligen Nutzer- bzw. Nutzungsprofils und das betriebsorganisatorische Umfeld heben gewisse DIRectory-Strukturen als praktisch sinnvoll gegenüber den individuellen und zufälligen Strukturierungen hervor.
Dabei ist vorrangig zu beachten, daß neben den Bestandteilen des **DOS-Betriebssystems** und der verwendeten Standardsoftware im Laufe der Zeit ständig neue Programme und Daten dem System – das heißt der konkreten PC-Installation – hinzugefügt werden, die entsprechend einzuordnen sind.
Diese Einordnung sollte aber in einem vorab festgelegten Konzept erfolgen und nicht operativ pragmatisch, was sehr bald zu einer chaotischen DIRectory-Struktur führt, bei der die Anzahl der SUB-DIRectories in Wildwuchs nach einer zufälligen Mnemonik ständig zunimmt. Parallel dazu wird dann meist auch die Sicherungsstrategie konzeptlos.
Im vorliegenden Artikel wird nun ein Ordnungskonzept vorgestellt, das auf einer Aufteilung der **Festplatte** in wahlweise ein oder zwei oder mehr logische Laufwerke (Laufwerk C und Laufwerk D) und auf den Laufwerken in sogenannte **Bibliotheken** (**PRO**Cedure **LIB**rary, **C**ommand **L**ibrary, **S**ource **L**ibrary) beruht.
Das Laufwerk C ist dabei dem eigentlichen DOS-Betriebssystem („Projekt"-Kurzzeichen: SYS1) zugeordnet. Auf dem Laufwerk D usw. werden weitere „Projekte" („Projekt"-Kurzzeichen: SYS2 ... SYSn) installiert, wobei die untergeordnete DIRectory-Struktur aller SYS2 ... SYSn-Systeme analog zur Struktur des [Betriebs-]Systems SYS1 auf dem Laufwerk C ist.
Falls die Kapazität der Festplatte 32 MByte nicht übersteigt, kann auf ein weiteres Laufwerk D usw. verzichtet und das ganze Bibliothekssystem auf dem Laufwerk C untergebracht werden.
Softwaremäßig eingerichtet wird das System SYS1 auf der Festplatte im logischen Laufwerk C bei der softwaremäßigen Vorbereitung des Computers durch einen kompetenten Systemorganisator (Systemprogrammierer). Unterstützt kann das werden durch entsprechende Generierungs-Tools. Das Verhältnis von Hardware zu Software ist zur Zeit so, daß Hardware als mehr oder weniger leeres „Gefäß" zur Aufnahme von Software bereitgestellt wird. Leider wird oft der Systemgenerierung bzw. Systeminstallation eines PCs zu wenig Aufmerksamkeit geschenkt und der Endnutzer eines PCs mit Fragmenten abgespeist, wo ein Gesamtkonzept notwendig wäre.
Das Organisationskonzept für die Festplatte mit den drei **Bibliothekstypen: PROCLIB, CL** und **SL** ist im Bild 1 gezeigt und wird nachfolgend beschrieben.

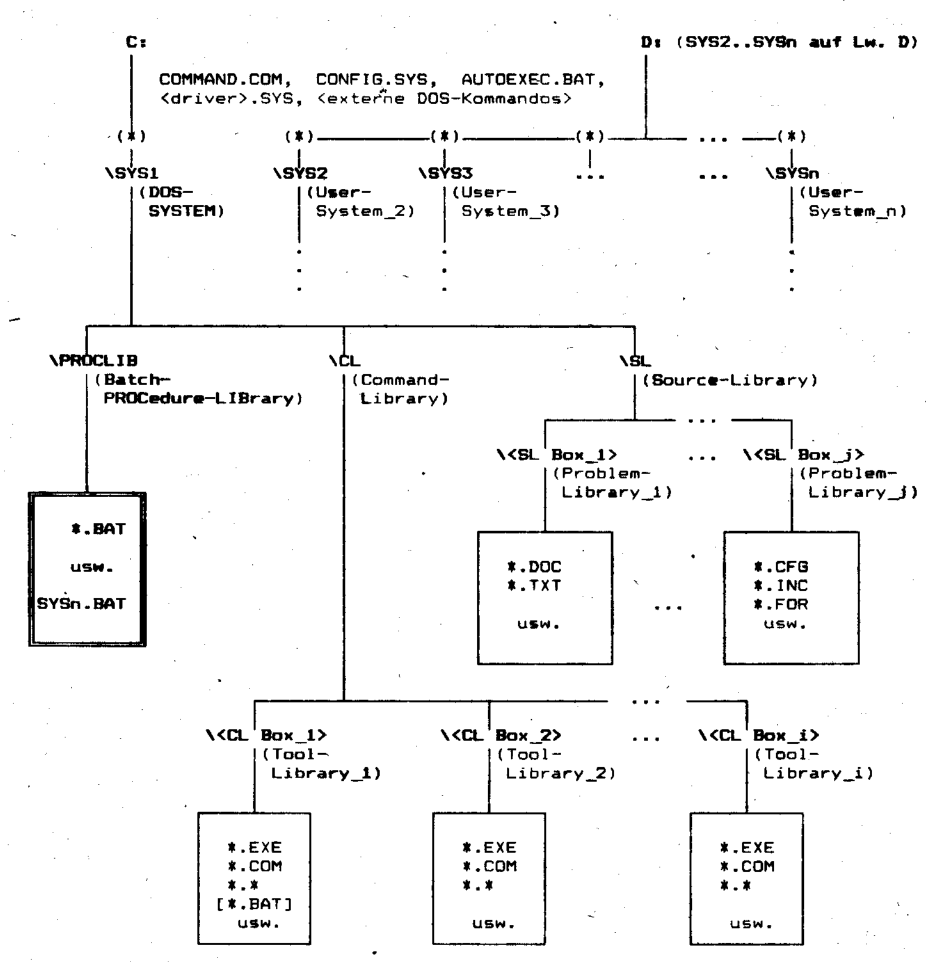
Das System „C:\SYS1\"

Im Laufwerk C der Festplatte ist im ROOT-Verzeichnis neben den für den Systemstart notwendigen Dateien (COMMAND.COM, CONFIG.SYS, AUTOEXEC.BAT, <driver>.SYS, <externe DOS-Kommandos> usw.) ein Unterverzeichnis **\SYS1** vorhanden. Dieses Verzeichnis gliedert sich in **drei weitere SUB-DIRectories**:

C:\SYS1\PROCLIB	→ Batch-PROCedure-LIBrary
C:\SYS1\CL	→ Command-Library
C:\SYS1\SL	→ Source-Library

Der Inhalt der Verzeichnisse **C:\SYS1\CL** bzw. **C:\SYS1\SL** sind entweder direkt Dateien vom Typ **EXE**, **COM** oder **BAT** u. a. oder üblicherweise weitere **SUB-DIRectories** einzelner Software-Tools als Zusammenfassung aller zu diesem bestimmten Software-Tool gehörenden Komponenten. Die unterste hierarchische Ebene ist damit die Stufe 3.
Praktische Erfahrungen zeigen, daß weitere Untergliederungen unterhalb der Stufe 3 zu einem Ordnungsumkehreffekt führen, das heißt, die erreichte Übersichtlichkeit im einzelnen tendiert wieder zu einer Unübersichtlichkeit im Ganzen.
Zum Beispiel seien alle zum Programmentwicklungssystem **TURBO**-Pascal Version 3.0 gehörenden Komponenten als Ergebnis der Übernahme und Generierung von den Distributionsdisketten für die vorliegende PC-Installation im Verzeichnis:
C:\SYS1\CL\TURBO3
enthalten.
Wahlweise werden nur für den gelegentlichen Gebrauch benötigte Komponenten – zum Beispiel Dokumentationen und Infor-

Bild 1 Festplattenorganisation und Bibliothekskonzept (PROCLIB, CL, SL)



mationen (README) u. a. – in dem Verzeichnis:
C:\SYS1\SL\TURBO3
gespeichert.
Die vorliegende Systemarchitektur setzt als „Kommando"-Zentrale der Stufe 1 das Batch-PROCedure-LIBrary-Verzeichnis:
C:\SYS1\PROCLIB
voraus.
In diesem Verzeichnis sind für alle Software-Tools und ausgewählte Programme – zum Beispiel FORMAT.EXE – Prozeduren (BAT-Dateien) vorhanden, die den Aufruf und die Arbeit mit dem entsprechenden Software-Tool gemäß der vorliegenden Installation voreinstellen; eventuell über eine Menusteuerung den Nutzer führen und damit eine Benutzeroberfläche gemäß dem speziellen Nutzer- und Nutzungsprofil schaffen. Im ENVIRONMENT ist standardmäßig zum Beispiel voreingestellt:
PATH=C:\;C:\SYS1\PROCLIB;C:\SYS1\CL\TOOLS
Durch den Aufruf von:
D>TURBO3<CR>
wird über den Pfad: **C:\SYS1\PROCLIB**
die Prozedure **TURBO3.BAT** aufgerufen, die unter anderem in das Verzeichnis **C:\SYS1\CL\TURBO3** führt.
Prinzipiell sind alle Software-Tools auf der Stufe 1 so zu installieren, daß sie in der **C:\SYS1\PROCLIB** als Prozedur verankert sind und statt als EXE-/COM-File über eine BAT-Datei aufgerufen werden, die auf das Nutzer-/Nutzungsprofil abgestimmt ist.

Die Systeme „D:\SYSn\"

Während das logische Festplattenlaufwerk C eine DIRectory-Struktur aus Bibliotheken für das DOS-Betriebssystem enthält, werden alle **Nutzer-Systeme** – als **„Projekte"** bezeichnet – auf das logische Festplattenlaufwerk D usw. gelegt. Ein Nutzersystem oder „Projekt" ist dabei gewissermaßen eine installationsbezogene Erweiterung des Systems SYS1, das heißt, im System SYS1 sind alle zentralen Komponenten der betreffenden Installation enthalten; in den Systemen **SYS2...SYSn** die nach „Projekten" oder Sachgebieten, Abteilungen, Mitarbeitern usw. gegliederten „Erweiterungen" des Systems SYS1. Damit wird eine OPEN-End-Struktur der Festplattenorganisation nach einheitlichem Bibliothekskonzept impliziert.
Wesentlich ist nun das Prinzip, die Systeme SYS2...SYSn auf dem Laufwerk D usw. mit einer zum System SYS1 analogen DIRectory-Struktur (Bibliotheksstruktur) aufzubauen. Damit erreicht man eine **Standard-System-Architektur**, die es erlaubt, zum Beispiel mit der Datei
C:\SYS1\PROCLIB\SYSx.BAT
den Anschluß des **Nutzersystems SYSx** durch Aufruf von:
D>SYSx<CR>
herzustellen.
Dabei wird unter anderem mit:
set path=D:\SYSn\PROCLIB;
D:\SYSn\CL\TOOLS;%PATH%
das Nutzersystem **SYSn** den Aufrufpfad betreffend vor das **Betriebssystem SYS1** gesetzt, so daß zum Beispiel eine eventuell abweichende TURBO3-Installation des Nutzersystems benutzt werden kann.
Die Verankerung des „Projekt"-Kurzzeichens SYSn im System SYS1 (SYSn.BAT) ist ein wesentlicher Punkt und zeigt die Priorität des Systems SYS1 über die Nutzersysteme SYS2...SYSn. Falls die Frage nach der Auswahl des „Projektes" SYSn in die AUTOEXEC.BAT aufgenommen wird, kann nach dem Systemstart sofort auf die entsprechende Nutzerebene und -oberfläche umgeschaltet werden.
Mit einer Prozedur SYS1.BAT der C:\SYS1\PROCLIB zum Beispiel kann man wieder auf die SYS1-Ebene gelangen.
Über die globale Variable PKZ („Projekt"-Kurz-Zeichen) beispielsweise kann man durch: **set PKZ=SYSn** das jeweilige angeschlossene „Projekt" im ENVIRONMENT verankern.
Das kann vor allem für verschiedene Versionen eines Software-Tools oder Projektes von Interesse sein. Die Benutzeroberflächen in den Systemen **SYSn** können durch unterschiedliche Inhalte der **D:\SYSi\PROCLIB** dem Nutzer-/Nutzungsprofil gezielt angepaßt werden.
Der Dreh- und Angelpunkt des Konzepts liegt in der Gestaltung der BAT-Dateien der **\SYSi\PROCLIB** und der Erkenntnis, daß die der PC-Installation zuzuführende Software nach einem einheitlichen Konzept installiert und eingebracht werden muß.
Da letzteres oft aus Gründen des Personal- und Zeitmangels nur unzureichend geschieht, treten immer wieder Effektivitätsverluste und Schlimmeres bei der Routinenutzung einer PC-Installation auf.

RAM-Disk und „\PROCLIB"

Da nach der hier zugrunde liegenden System-Architektur als **„Kommando"-Zentrale** die PROCedure LIBrary auf der Harddisk einem ständigen Zugriff ausgesetzt ist, sollte man durch Einrichten einer RAM-Disk und Kopieren der C:\SYS1\PROCLIB sowie der D:\SYSn\PROCLIB in diese **schnelle „Diskette"** die Arbeit mit den BAT-Dateien beschleunigen. Temporäre Veränderungen an den Inhalten der Prozeduren im Verlaufe der Arbeit können dann ebenfalls in der RAM-Disk vorgenommen werden, wodurch die Dynamik der Architektur eine wesentliche Erweiterung erfährt. Bei Anwendung von FORTRAN, dBASE III-Plus oder TURBO-Pascal Version 4.0 u. a. ist bei einem Rechner mit nur 512 KByte RAM-Speicherplatz die Arbeit mit einer RAM-Disk sehr eingeschränkt oder unmöglich, da ja bekanntlich der Platz für die RAM-Disk vom verfügbaren RAM-Speicherbereich abgezogen wird. Sicherlich reichen etwa 100 KByte RAM-Disk für die Aufnahme der BAT-Dateien aus.

Literatur
/1/ Weber, R.: So beugen Sie dem Chaos auf der Festplatte vor.
Computerheft (1987) 1, S. 27
/2/ MS-DOS in der Praxis.
MS-DOS Welt (1987) 1, S. 11

✉ KONTAKT ☎
Technische Universität Dresden, Sektion Wasserwesen, Mommsenstraße 13, Dresden, 8027; Tel. 2 32 61 18

ROLANET 1 mit Lichtwellenleitern

Dr. Andreas Barsch,
Dr. Karl-Heinz Jänicke
Humboldt Universität zu Berlin, Sektion Elektronik

Das Vernetzen von Computern hat sich in den letzten Jahren als eine besonders dringliche Aufgabe für viele gesellschaftliche Bereiche erwiesen. Oft besteht die Notwendigkeit, die Lösung von Aufgaben durch Dezentralisation der Bearbeitung bei gegebener Zugriffsmöglichkeit auf zentrale Ressourcen zu finden. Eine andere Form ist die Ensemble-Bearbeitung komplizierter Probleme durch mehrere Computer mit relativ geringer Leistungsfähigkeit. Auch das Bedürfnis nach Informationsaustausch untereinander begründet eine Computervernetzung. Diesen Tendenzen folgend, wurde vom VEB Kombinat Robotron mit der Entwicklung des lokalen Rechnernetzes (LAN) ROLANET 1 ein erster Schritt getan. Perspektivisch soll ROLANET 1 die Verkopplung aller Rechner dieses Kombinates ermöglichen /1/, /2/.

Lichtwellenleiter im LAN
Ausgangspunkt

Lokale Rechnernetze mittlerer Geschwindigkeit (0,5...4 MBit/s) bilden weltweit den überwiegenden Teil der eingesetzten LANs. Die übertragungstechnischen Komponenten (Kabel, Transceiver, Repeater, ...), die rechentechnischen Komponenten (LAN-Controller, Server, ...), die Basissoftware zur Abwicklung der Datentransportfunktionen und die Anwendersoftware zur Realisierung der gewünschten Applikationen stellen die Hauptbestandteile eines solchen Neztes dar. Bei der Strukturierung dieser Komponenten liegt meist das OSI-Referenzmodel der Internationalen Standardisierungsorganisation (ISO) zugrunde. Folglich bilden die ISO-Standards für die einzelnen Schichten des Referenzmodells zunehmend den Ausgangspunkt für konkrete Implementationen. Ein LAN stellt sich somit als Ensemble wohlabgestimmter Hardware- und Softwarekomponenten dar, dessen Komplexität bei Integration verschiedener Rechner in das Netz noch zunimmt.
Für jeden anzuschließenden Rechner müssen die LAN-Controller, auch Network Interface Unit (NIU) genannt, entwickelt und produziert werden. Diese LAN-Controller umfassen meist eine Leiterkarte und sind auf die

Hardware des Rechners abgestimmt. Die Basissoftware muß dem jeweiligen Controller und Rechner angepaßt sein. Die Bereitstellung der letztlich entscheidenden Komponente, der erforderlichen Applikationssoftware, die in engem Zusammenhang mit dem Betriebssystem des jeweiligen Rechnertyps steht, verursacht wesentliche Aufwendungen beim LAN-Produzenten bzw. beim Anwender. Lediglich die übertragungstechnischen Komponenten sind unabhängig von den anzuschließenden Rechnern, weil ein einheitliches Interface zwischen Transceiver, auch Medium Attachment Unit (MAU) genannt, und LAN-Controller der verschiedenen Rechner existiert. Erst die Lösung aller genannten Aufgaben und die (eventuell auch arbeitsteilige) Bereitstellung der verschiedenen LAN-Komponenten ermöglichen die breite Anwendung des lokalen Rechnernetzes und die Erzielung entsprechender Rationalisierungseffekte.
Der Aufwand zur Produktion eines vollständigen Vernetzungsinstrumentariums für verschiedene vorhandene Rechner ist also außerordentlich hoch. Die Orientierung auf mehrere zueinander nicht vollständig kompatible lokale Rechnernetze einer Geschwindigkeitsklasse für das gleiche Aufgabenspektrum erscheint daher volkswirtschaftlich nicht vertretbar.

Lösungsansatz

Die Anwendung von Lichtwellenleitern (LWL) in LAN mittlerer Geschwindigkeit sollte aus den oben genannten Gründen möglichst wenige neue Komponenten erfordern und auf ein vorhandenes Netzwerkkonzept aufbauen. So können einige der vorteilhaften Eigenschaften des Lichtwellenleiters, wie z. B. die elektromagnetische Unempfindlichkeit, die geringe Dämpfung und die Abhörsicherheit, bei Nutzung eines vorhandenen Vernetzungsinstrumentariums voll zur Wirkung kommen. Ein wesentlicher Aspekt des sonst üblichen LWL-Einsatzes, die hohe übertragbare Bitrate, bleibt in diesen Netzen notwendigerweise ungenutzt.
Der Weg der LWL-Anwendung besteht also in einer teilweisen oder vollständigen Substitution der übertragungstechnischen Komponenten. Die konkreten Definitionen der übertragungstechnischen Schnittstellen und Mechanismen von ROLANET 1 bilden hier die Grundlage für die Entwicklung der optischen Komponenten und für die Wahl der Funktionsprinzipien.

ROLANET 1

LAN-Controller für die jeweiligen Rechnertypen (LNC1-XXXX) und ein einheitlicher Transceiver (TCR) bilden die Hardwarekomponenten von ROLANET 1 (Bild 1). Der LAN-Controller ist in seiner Anwendung Bestandteil des Rechners, wogegen der Transceiver als Element der übertragungstechnischen Komponenten in einer Entfernung von bis zu 50 m vom Rechner entfernt plaziert werden darf. Das Transceiver-Kabel verbindet beide Baugruppen. Die Transceiver der einzelnen zum LAN gehörenden Rechner (Netzrechner, NR) sind wiederum über ein gemeinsames Koaxialkabel in Bustopologie miteinander verbunden. Das kollisionsbehaftete CSMA/CD-Verfahren (Carrier Sense Multiple Access/Collision Detection) bildet die Grundlage für die Zugriffssteuerung. Die Daten werden mit einer Geschwindigkeit von 0,5 MBit/s übertragen.
Der LAN-Controller stellt eine mikroprozessorgestützte Interfacekarte dar. Er entlastet den Rechner von einer Reihe datenübertragungstypischer Aufgaben (Senden, Empfangen, Fehlerkontrolle, Zugriffssteuerung auf das Übertragungsmedium,...). Der interne Rechnerbus und der Transceiver-Anschluß bilden die Schnittstellen.
Die Transceiver übernehmen die Weiterleitung und Verstärkung der Signale vom Koaxialkabel zum Rechner und umgekehrt sowie das Erkennen und Melden einer vorliegenden Kollision auf dem Koaxialkabel. Das Koaxialkabel selbst ist durch die Transceiver ohne Unterbrechung durchgeschleift, so daß die rückwirkungsfreie An- und Abkopplung eines Rechners an das LAN sichergestellt wird. Die Spannungsversorgung (+12 V) des Transceivers erfolgt vom angeschlossenen Rechner über das Transceiver-Kabel.
Das Koaxialkabel darf maximal 1 000 m lang sein. Die notwendige Kollisionserkennung verursacht wesentlich diese Begrenzung. Es muß sichergestellt werden, daß ein gleichzeitiges Senden der entferntesten Rechner zuverlässig durch die beteiligten Transceiver erkannt wird. Die Zeitbedingungen des implementierten CSMA/CD-Zugriffsverfahrens erlauben aber eine erheblich größere Netzausdehnung.
Das Transceiver-Interface umfaßt dagegen mehrere Signalleitungen (Tafel 1). Sendedaten werden dem Transceiver über das Leitungspaar **Transmit** zugeleitet. Die Empfangsdaten erreichen den LAN-Controller über die Signalleitungen **Receive**. Eine beim Senden aufgetretene Kollision erkennt der Controller an einem aktiven Signal **Collision**, worauf er diese Sendung determiniert abbricht und nach vorgeschriebenen Algorithmen einen neuen Sendeversuch einleitet. Das implementierte CSMA/CD-Zugriffsverfahren widerspiegelt sich also an dieser Schnittstelle. Dabei ist es z. B. uninteressant, auf welchen physikalischen Prinzipien beruhend eine Kollision erkannt wird. Lediglich die Aktivierung und Deaktivierung der Signale nach den vorgeschriebenen statischen und dynamischen Bedingungen muß gewährleistet werden.
Das Transceiver-Interface und das Koaxialkabel, einschließlich der darauf ablaufenden Vorgänge, bilden die denkbaren Schnittstellen für den Einsatz optischer Komponenten in ROLANET 1.

Tafel 1 Signalauswahl Transceiver-Interface

Signal	Bezeichnung	Kontakt	Bemerkung
Sendesignal (Transmit Pair)	Transmit + Transmit –	(3) (10)	symmetrische Leitung mit V11 Kabelsender und -empfänger
Empfangssignal (Receive Pair)	Receive + Receive –	(5) (12)	"
Kollisionssignal (Collision Pair)	Collision + Collision –	(2) (9)	"
Spannung, 12P Masse, 00 (Power Pair)	Power + Power –	(13) (6)	

Optisches CSMA/CD-LAN
Die optischen Komponenten

Eine Analyse verfügbarer optischer Basiselemente (Sendedioden, Empfangsdioden, Verzweigungselemente, Lichtwellenleiter, Steckverbinder,...) ergab, daß der Aufbau optischer Busstrukturen mit größerer Anschlußzahl ohne Integration von Verstärkern nicht möglich ist. Zusätzlich erschwert das verwendete kollisionsbehaftete CSMA/CD-Zugriffsverfahren generell eine Anwendung optischer Busstrukturen auf der Basis passiver optischer Verzweigungselemente /4/. Der Entwurf optischer Strukturen unter Nutzung einzelner Punkt-zu-Punkt-Übertragungen mit Lichtwellenleitern stellt dagegen einen Ausweg dar.
Vier verschiedene optische Komponenten mit einer speziellen optischen Schnittstelle wurden zur Lösung dieser Aufgabe entwikkelt. Die optische Schnittstelle verfügt über je einen Anschluß für einen zuführenden und einen wegführenden Lichtwellenleiter. Ein spezielles Schaltverhalten sichert die Weiterleitung einer aufgetretenen Kollision. Das über den zuführenden LWL empfangene Datenpaket wird an alle anderen Interfaces der optischen Komponente weitergeleitet, nur nicht an den wegführenden LWL. Die optische Schnittstelle arbeitet also in diesem Sinne reflexionsfrei. Nur im Kollisionsfall zwischen den Interfaces der entsprechenden optischen Komponente wird diese Reflexionsfreiheit aufgehoben.
Ein **optischer Transceiver (OTCR)** erzeugt den Übergang vom ROLANET 1-Transceiver-Interface zu der beschriebenen optischen Schnittstelle. Die Spannungsversorgung (+12 V) erfolgt, wie beim elektrischen Vorbild, vom angeschlossenen Rechner über das Transceiver-Kabel. Der OTCR umfaßt eine Leiterkarte mit den Maßen 105 mm × 200 mm und findet in einem Gehäuse mit den Abmessungen des elektrischen Transceivers Platz.
Der **aktive optische Sternkoppler (AOS)** verfügt über maximal 16 optische Schnitt-

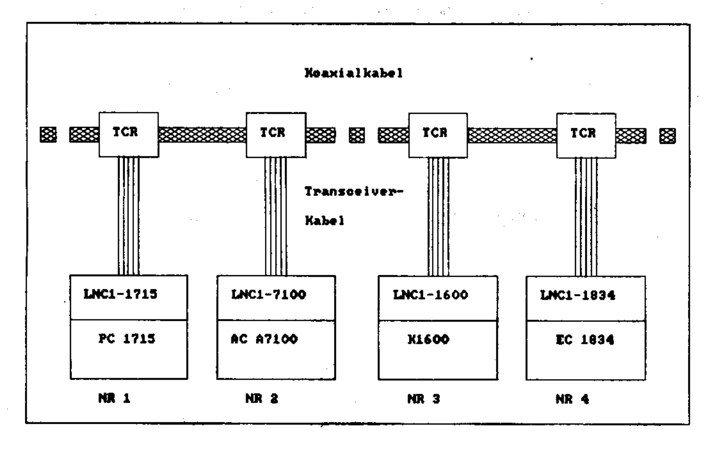


Bild 1 ROLANET-1-Konfiguration

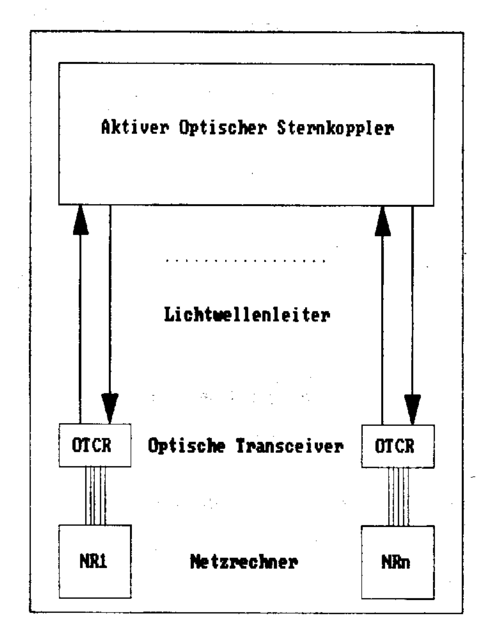


Bild 2 Grundkonfiguration Optisches ROLANET 1

Bild 3 Erweiterte Variante Optisches ROLANET 1

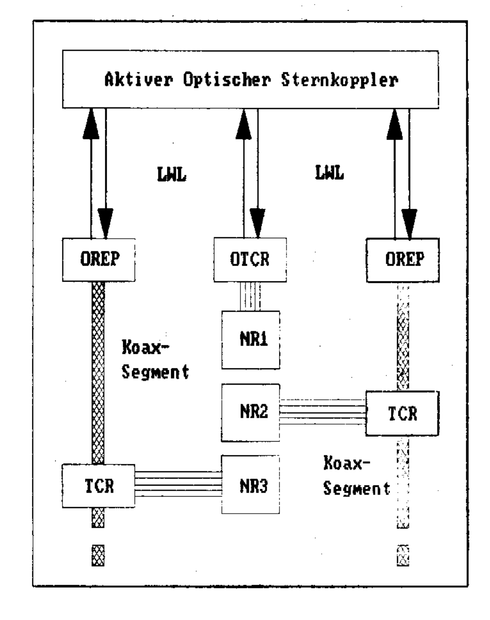


Bild 4 Koax-Segment-Anschluß an Optisches ROLANET 1

stellen der oben genannten Art. Er dient somit der Weiterleitung eines ankommenden Datenpaketes von einer optischen Schnittstelle an alle anderen. Der AOS besitzt eine eigene Stromversorgung und nutzt ein EGS-Gehäuse. Die optischen Schnittstellen umfassen je eine Leiterkarte mit den Maßen 95 mm × 170 mm. Eine Steuerplatine mit den gleichen Abmessungen unterstützt das Schaltverhalten der optischen Schnittstellen.
Der Übergang vom ROLANET-1-Koaxialkabel auf das optische Interface wird von dem **optischen Repeater (OREP)** geschaffen. Sein Koaxialkabelanschluß entspricht dem des elektrischen Transceivers. Auch er nutzt ein Gehäuse mit den Abmessungen des elektrischen Transceivers. Die Spannungsversorgung (9...12 V Wechselspannung) erfolgt über einen separaten Transformator.
Der **optische Multitransceiver (OMTCR)** trägt der Tatsache Rechnung, daß sich in konkreten Anwendungsfällen meist mehrere Computer in einem Raum befinden. Ihre Verkopplung über Lichtwellenleiter oder auch über Koaxialkabel wäre zu aufwendig. Der OMTCR löst dieses Problem auf einfache Art. Vier vorhandene, zu ROLANET 1 kompatible Transceiver-Interfaces gestatten bereits die Kopplung von vier Rechnern und die Nutzung der ROLANET-1-Vernetzungsinstrumentarien. Der OMTCR unterstützt dabei das CSMA/CD-Zugriffsverfahren. Die gesendeten Datenpakete eines angeschlossenen Rechners werden auf die anderen Transceiver-Interfaces und auf die optische Schnittstelle der oben beschriebenen Art geleitet. Die optische Schnittstelle ermöglicht die direkte Kopplung mit einer beliebigen anderen optischen Komponente.
Der Stufenprofillichtwellenleiter und die in einer früheren Arbeit entwickelten optischen Sende- und Empfangsmoduln (SM 1100, EM 1002) bilden die Grundlage für die optischen Komponenten /5/. Optische Strecken von mehr als 1,5 km Länge können so unter Berücksichtigung der Kabeldämpfung überbrückt werden. Bei geringeren Forderungen an die Netzausdehnung und damit an die überbrückbare Streckenlänge sowie an die verursachten Tastgradverzerrungen durch die optischen Moduln sind auch die jetzt verfügbaren, industriell gefertigen Sende- und Empfangsmoduln OS 500 und OE 500 vom VEB Kombinat EAW einsetzbar.

Realisierbare Netztopologien

Die optischen Komponenten erlauben den Aufbau vielfältiger, den konkreten Gegebenheiten optimal angepaßter Netztopologien unter voller Nutzung aller ROLANET-1-Hardware- und Softwarekomponenten. Insbesondere unterstützt die definierte optische Schnittstelle eine direkte Kopplung beliebiger optischer Komponenten untereinander.
Der optische Sternbus unter Verwendung des aktiven optischen Sternkopplers stellt die Grundkonfiguration des optischen CSMA/CD-LANs bei Nutzung von ROLANET-1-Baugruppen dar. Bild 2 zeigt die Anordnung. Die einzelnen Netzrechner mit den rechnerinternen LAN-Controllern sind über das Transceiver-Kabel an die optischen Transceiver geschaltet. LAN-Controller und Transceiver-Kabel gehören dem bisherigen ROLANET-1-System an. Die optischen Transceiver werden mittels Lichtwellenleitern über den aktiven optischen Sternkoppler miteinander verbunden. Die Kopplung von insgesamt 16 Netzrechnern ist mit einer solchen Anordnung möglich.
Besteht die Notwendigkeit zur Vernetzung von mehr als 16 Rechnern, so ergeben sich auf der Basis der entwickelten Baugruppen zwei prinzipielle Lösungswege. Bild 3 zeigt beide Möglichkeiten. Der erste Weg besteht in der Anwendung des optischen Multitransceivers. Er nutzt die Tatsache, daß meist mehrere Rechner dicht beieinander stehen. Eine Kopplung untereinander über Lichtwellenleiter ist dann weder technisch notwendig noch ökonomisch gerechtfertigt. Der Multitransceiver ermöglicht die Kopplung dieser Rechner im Nahbereich (maximal 50 Meter) über die elektrische Transceiver-Schnittstelle. Weiterhin sind die Rechner aber gleichzeitig mit dem gesamten Übertragungssystem des LAN über das Lichtwellenleiterpaar zum Sternkoppler verbunden. Die Anzahl verkoppelbarer Netzrechner erhöht sich damit auf 64.
Ein zweiter Weg zur Vergrößerung dieser Anzahl besteht in der sogenannten Kaskadierung der optischen Sternkoppler. Die verwendeten Mechanismen zur Unterstützung des CSMA/CD-Verfahrens (Reflexionsfreiheit der optischen Schnittstelle) erlauben eine Kaskadierung auf sehr einfache Art. Ein zweiter optischer Sternkoppler wird über ein Lichtwellenleiterpaar an den ersten Sternkoppler in der gleichen Art und Weise angeschaltet wie ein Transceiver und umgekehrt. Neben der Erhöhung der Zahl anschließbarer Rechner ergibt sich ein zweiter Effekt – die Erweiterung der Ausdehnung. Zwischen zwei Sternkopplern kann dann wieder eine Strecke entsprechend der Reichweite der verwendeten optischen Sende- und Empfangsmoduln überbrückt werden. Es besteht auch die Möglichkeit, sofern die Ausdehnungserweiterung nicht erforderlich ist, diese Sternkoppler über elektrische Leitungen im Nahbereich zu kaskadieren. Maximal vier Sternkoppler können zwischen zwei beliebige Netzrechner geschaltet sein.
Der Aufbau von lokalen Rechnernetzen auf der Basis von Lichtwellenleitern ist natürlich nur dort gerechtfertigt, wo auch die Vorteile dieses Übertragungsmediums zur Geltung kommen. In Bereichen, in denen das Koaxialkabel allen Ansprüchen gerecht wird und deshalb auch ökonomisch nur zu vertreten ist, existieren dann ROLANET-1-Systeme, auch als Koax-Segmente bezeichnet. Es entstehen die Forderungen nach Kopplung dieser Segmente zu einem einheitlichen Rechnernetz oder auch nach Ankopplung von Koax-Segmenten an ein optisches LAN. Diese Kopplung übernimmt ein optischer Repeater. Bild 4 zeigt eine solche prinzipielle Lösung.
Ein optisches LAN mit dem aktiven optischen Sternkoppler als wesentliche Komponente bildet die Grundlage. An diesen Sternkoppler werden nun über optische Repeater (OREP) die ROLANET-1-Koax-Segmente geschaltet. Jedes Segment kann die maximale Länge von ROLANET 1 annehmen und jeweils die vertretbare Zahl Netzrechner versorgen. In Bild 4 ist Netzrechner 3 über einen elektrischen Transceiver an das erste Seg-

ment geschaltet, und Netzrechner 2 wirkt als Nutzer im zweiten Koax-Segment. Der aktive optische Sternkoppler ist in der Lage, weitere Elemente eines Übertragungssystems zu koppeln. So besteht die Möglichkeit, Netzrechner über optische Transceiver anzuschalten, wie im Bild 4 dargestellt. Die Ankopplung weiterer Koax-Segmente über optische Repeater sowie die Verbindung mit anderen Sternkopplern oder Multitransceivern ist ebenfalls möglich. Es stehen somit Instrumentarien zur Verfügung, die eine optimale Anpassung des Übertragungssystems an topologische Forderungen und Gegebenheiten gestatten. Lediglich die Gesamtzahl der zwischen zwei beliebigen Netzrechnern plazierten optischen Baugruppen muß dabei berücksichtigt werden.

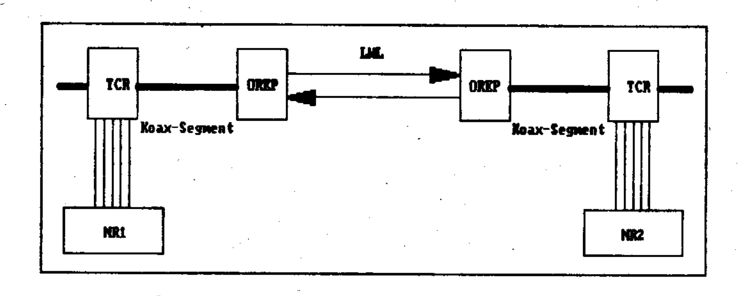
Neben diesen komplexen Netzkonfigurationen ergeben sich auch einfache Lösungen zur partiellen Anwendung der optischen Komponenten. Bild 5 zeigt einige Beispiele. Zwei optische Repeater erlauben die Kopplung zweier Koax-Segmente über eine Entfernung entsprechend der Systemleistung der eingesetzten optischen Sende- und Empfangsmoduln. Die Anwendung dieser Lösung erscheint gerade dann vorteilhaft, wenn z. B. zwei in voneinander entfernten Gebäuden befindliche Koax-Segmente miteinander verbunden werden sollen. Das zweite Beispiel zeigt die Anwendung eines optischen Repeaters und eines optischen Multitransceivers. Bis zu vier vom elektrischen LAN entfernte Rechner können auf diese Weise in das LAN integriert werden. Diese Forderung entsteht meist bei Einbindung von Ressourcen eines entfernten Rechenzentrums in ein vorhandenes ROLANET 1. Zwei optische Multitransceiver können aber auch bereits die Verdopplung von bis zu acht Rechnern in der zuletzt gezeigten Art übernehmen. Jeweils vier dieser Rechner sind über die Transceiver-Kabel und den OMTCR miteinander verbunden und erlauben bereits die Kommunikation über ROLANET-1-Software. Die Verbindung beider OMTCR mittels zweier Lichtwellenleiter erweitert diese Kommunikationsmöglichkeit auf alle acht Rechner.

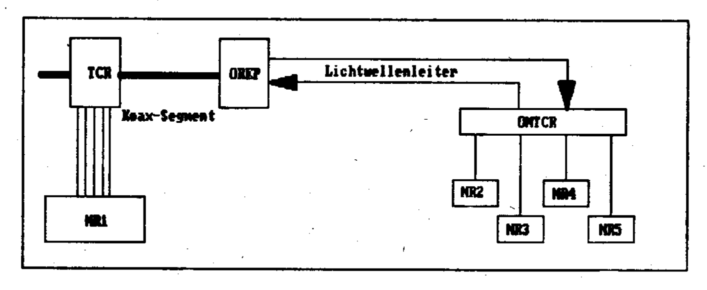
Zusammenfassung

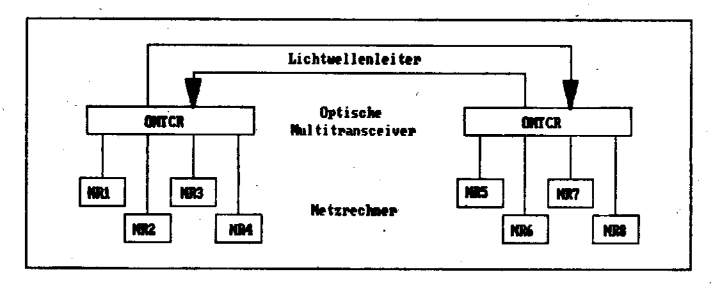
Die Anwendung von Lichtwellenleitern in lokalen Rechnernetzen mittlerer Geschwindigkeit erfordert keine Neuentwicklung der rechentechnischen Hardware. Vielmehr ist es möglich, in ein vorhandenes lokales Rechnernetz (ROLANET 1) optische Komponenten zu integrieren. Diese Nutzung der optischen Komponenten hat keinen Einfluß auf die zur Verfügung stehende Firmware des LAN. Es erfolgt lediglich eine teilweise oder vollständige Substitution der elektrischen Komponenten des Übertragungssystems. Die optischen Komponenten Sternkoppler, Transceiver, Multitransceiver und Repeater erlauben vielfältige Möglichkeiten zur Gestaltung der Topologie von ROLANET 1. Rechnernetze mit ausschließlich optischen Übertragungsmitteln und gemischte Konfigurationen aus Lichtwellenleitern und Koaxialkabeln lassen sich aufbauen.

Dem lokalen Rechnernetz ROLANET 1 werden auf der Basis der vorgestellten Lösungen neue Anwendungsgebiete erschlossen. Sie resultieren aus der Erweiterung der Ausdehnung des Rechnernetzes und aus dem Einsatz dieses Netzes in Bereichen, wo die Anwendung des Koaxialkabels aus verschiedensten Gründen bisher ausgeschlossen war.

Bild 5 Minimalkonfigurationen Optisches ROLANET 1







Literatur

/1/ Terpe, B.: ROLANET 1 – Eine Konzeption für lokale Netze des VEB Kombinat Robotron. Symposium „Das lokale Rechnernetz ROLANET", Dresden, 19. Juni 1987

/2/ Richter, J.; Terpe, B.: ROLANET 1 – ein lokales Netz des VEB Kombinat Robotron. Neue Technik im Büro 30 (1986) 5, S. 146

/3/ Heymer, V.: Lokale Rechnernetze mit OSI-Architektur. Mikroprozessortechnik 1 (1987) 3, S. 74

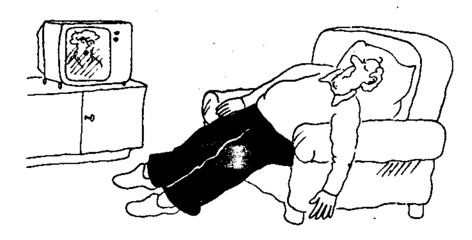
/4/ Barsch, A.: Lichtwellenleiter kontra CSMA/CD? Mikroprozessortechnik 2 (1988) 6, S. 183

/5/ Zaremba, J.; Jänicke, K.-H.: Lokales Rechnernetz LAN-CELOT 1. Nachrichtentechnik Elektronik 35 (1985) 7, S. 242

✉ KONTAKT ☎

Humboldt-Universität zu Berlin, Sektion Elektronik, Bereich Technische Informatik, Invalidenstraße 110, Berlin, 1040, Tel. 2 80 35 78 (Dr. Jänicke)

Bildschirmposition



Zeichnung: Frank Steger



Der Bezirksfachausschuß Mikroelektronik beim Bezirksvorstand Berlin der KDT führt im ersten Halbjahr 1989 Spezialkonsultationen zu folgenden Themen durch:

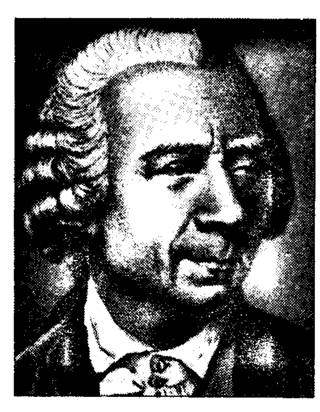
11. Januar	Mikroelektronik und flexible Fertigungsautomatisierung
8. Februar	Anwendungsbaugruppen zum EC1834
22. März	Busstrukturen in der Automatisierungstechnik
12. April	P 8000 Compact – Die Weiterentwicklung des P 8000
10. Mai	Projektierungstechnologien für Automatisierungsanlagen
21. Juni	Informationen zur Nutzung des automatischen Datennetzes der Deutschen Post

Änderungen sind dem Bezirksfachausschuß Mikroelektronik vorbehalten. Die Teilnahme an den Konsultationen bedarf keiner Anmeldung und ist kostenlos.

Die Konsultationen finden jeweils 14.00 Uhr im Haus der KDT, 1086, Berlin, Kronenstraße 18, statt.

Driese

Wegbereiter der Informatik



LEONHARD EULER

∗ 1707 Basel,
† 1783 St. Petersburg

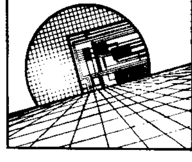
Auf jedem Schul-Taschenrechner sind Funktionstasten für die Exponentialfunktion und für die natürlichen Logarithmen vorhanden. Die Basiszahl für diese Funktionen ist die bekannte *Eulersche Zahl*

$$e = \lim_{n \to \infty} (1 + 1/n)^n.$$

Auf vielen Taschenrechnern gibt es auch eine Rechentaste für die Funktion n! („n Fakultät"):

$$n! = \prod_{k=1}^{n} k;$$

diese Funktion ist für jede natürliche Zahl n definiert; Euler fand, daß man diese Funktion auch für beliebige, positive reelle Werte erklären kann und gab als Lösung die in der Mathematik geläufige *Eulersche Gamma-Funktion* an.
Es gibt noch weitere Bezüge, durch die Eulers Name mit der modernen Rechen- und Programmiertechnik verbunden ist. Genannt sei z.B. das *Eulersche Streckenzugverfahren*, das einfachste 1-Schritt-Verfahren zur Lösung von gewöhnlichen Differentialgleichungen, das zudem immer numerisch stabil ist.
Die Aufzählung ließe sich fortsetzen, denn die Anzahl der mit dem Namen Eulers in Zusammenhang stehenden Begriffe in einem mathematischen Lexikon ist extrem hoch und nur noch mit der von Gauß vergleichbar.
In seiner Kindheit erhielt Euler ausschließlich Privatunterricht, und zwar hauptsächlich von seinem Vater, einem Pfarrer, der großes Interesse für Mathematik besaß und sogar bei dem berühmten Jacob Bernoulli studiert hatte. Bereits mit 13 Jahren wurde Euler an der philosophischen und später an der theologischen Fakultät der Universität Basel immatrikuliert, wo er nebenher mathematische Vorlesungen bei Johann Bernoulli hörte. Sein Vater war einsichtig genug, die für Leonhard vorgesehene Theologenlaufbahn zugunsten dessen frühzeitiger mathematischer Erfolge aufzugeben. Schon als 16jähriger beendete Euler die philosophische Fakultät mit einem Examen, das ihm die Magisterwürde einbrachte. Seinen Studienfreunden Daniel und Nikolaus Bernoulli folgend, reiste er 1727 nach St. Petersburg, wo er 1730 an der Akademie eine Professur für Physik und 1733 für Mathematik erhielt. Hier ging er auch mit der Tochter eines Schweizer Malers eine Ehe ein, aus der 13 Kinder hervorgingen (aber 8 frühzeitig verstarben).
In St. Petersburg übernahm Euler die Aufsicht über das Geographische Department, wurde Mitarbeiter in der Kommission für Maß und Gewicht und war an den Vorbereitungen der großen Kamtschatka-Expedition (1733–43) beteiligt. Sein Hauptwerk jener Zeit ist ein zweibändiges Buch der Mechanik, womit er etwas völlig Neues in der Wissenschaft einführte: das Schreiben von Lehrbüchern. Getrübt wurde Eulers Petersburger Zeit durch den Verlust der Sehkraft seines rechten Auges als Begleiterscheinung einer Infektion, die er selbst auf Überanstrengung zurückführte.
Nach dem Tode der Kaiserin Anna von Rußland wurde die Lage der russischen Akademie unsicher, und Euler folgte 1741 gern einem Ruf Friedrichs II. an die Berliner Akademie, wo er Direktor der mathematischen Klasse wurde. Auf sein Wohnhaus in der Berliner Behrenstraße 21 weist noch heute eine Gedenktafel hin. Die Berliner Zeit (25 Jahre) war für Euler sehr schaffensreich, er verfaßte 380 Arbeiten sowie einige Bücher. Hier vollendete er unter anderem eines seiner wichtigsten Werke, die Variationsrechnung (C. Carathéodory bezeichnete es als „eines der schönsten mathematischen Werke, die je geschrieben worden sind").
Lange Zeit verwaltete er die Akademie quasi als Vizepräsident und brachte sie – ebenso wie die Petersburger – in die erste Reihe der europäischen Akademien. Wegen Zerwürfnissen mit Friedrich II. (abgewiesene Akademie-Präsidentschaft; der König liebäugelte mit radikalen französischen Aufklärern, was dem tiefreligiösen Euler unerträglich war) zog Euler 1766 wieder nach St. Petersburg, wo er von Katharina II. in großen Ehren aufgenommen wurde. Kurz danach erblindete E. völlig, doch beeinträchtigte das seine Schaffenskraft in keiner Weise – vielmehr schien sich sein ganzes Genie jetzt voll zu offenbaren: Fast die Hälfte seiner Arbeiten entstand in der Zeit seiner Blindheit! Gestützt auf sein phänomenales Gedächtnis arbeitete er wissenschaftlich intensiv weiter, indem er seinem Sohn Albrecht (1734–1800) diktierte.
So ist in Eulers Leben ein außerordentlich umfangreiches Gesamtwerk entstanden: Es umfaßt 886 Titel; darunter befinden sich 40 Lehrbücher, deren Darstellungsform z.T. endgültig gewesen und von bedeutenden Mathematikern der nachfolgenden Zeit übernommen worden ist. Kein Geringerer als C. G. J. Jacobi (einer der bedeutendsten deutschen Mathematiker nächst Gauß) sowie P. H. Fuß (ein Urenkel Eulers) bemühten sich um eine Gesamtausgabe des Eulerschen Werkes, scheiterten aber an dem Umfang dieser Aufgabe. Aus Anlaß seines 200. Geburtstages wurde erneut eine Gesamtausgabe beschlossen, und so erschien 1911 (im Teubner-Verlag) der 1. Band der inzwischen auf über 70 Bände angewachsenen und vom Baseler Birkhäuser-Verlag weitergeführten Edition.
So umfangreich, wie Eulers Schaffen ist, so vielseitig ist es auch – einige Beispiele mögen dies demonstrieren. In der Petersburger Akademie gehörte Euler mehreren Kommissionen zur Lösung technischer und praktischer Fragen an. In Zusammenhang damit beschäftigte er sich unter anderem mit Feuerspritzen, Ofenkonstruktionen, der Saugwirkung von Pumpen, dem Schleusen- und Kanalbau, dem Entwurf idealer Zahnradprofile und erstellte ein Gutachten, wie die Riesenglocke in Moskau auf den Kremlturm gehoben werden könne. Er erdachte ein Verfahren, aus nur drei Beobachtungen eines Planeten dessen Bahn zu bestimmen. Er konnte auch eine bei Newton offen gebliebene Frage aus der Gezeitentheorie (über das Zurückbleiben der Flutwelle gegenüber der Kulmination des Mondes) klären und löste damit eine Preisfrage der Pariser Akademie. Des weiteren griff Euler eine Anregung von Leibniz auf, die dieser in einem Brief an Huygens geäußert hatte, und er befaßte sich mit Topologie: Er löste das bekannte Königsberger Brükkenproblem und dessen Verallgemeinerungen und fand den *Eulerschen Polyedersatz*. Bemerkenswert ist sein Versuch, „die Musik als Teil der Mathematik auszuführen". Selbst Klavierspieler, wollte Euler die Musik aus den sichersten Grundlagen der Harmonie ableiten und schrieb zu diesem Zweck den „Versuch einer neuen Musiktheorie" (1739), dem später noch drei weitere Abhandlungen folgten.
Auch hat er die Durchdringung des Lichts durch verschiedene Medien untersucht. Es ist kaum zu glauben: Das daraus entstandene Lehrbuch der geometrischen Optik („Dioptik", 1768) wurde von einem Blinden verfaßt! Sogar Fragen der Stabilität, des Gleichgewichts und des Schaukelverhaltens von Schiffen hat er untersucht und dafür 1759 den Preis der Pariser Akademie erhalten. 1773 veröffentlichte er eine vollständige Theorie des Schiffbaus und der Navigation – eine Arbeit, die auch in England, Frankreich und Italien herausgegeben wurde.
Dies alles in Betracht ziehend nimmt es nicht wunder, daß Euler schon zu Lebzeiten fast zur Legende wurde – man hat ihn sogar die „lebendige Analysis" genannt. In Würdigung seiner mathematischen Leistungen schrieb C. G. J. Jacobi, daß Euler in seiner Berliner Zeit die gesamte Mathematik umgestaltet habe. Und der französische Mathematiker P. S. Laplace pflegte zu sagen: „Lest Euler, er ist unser aller Meister". *Dr. Klaus Biener*



13. Mikroelektronik-Bauelemente-Symposium

Das 13. Mikroelektronik-Bauelemente-Symposium als gemeinsame Veranstaltung des VEB Kombinat Mikroelektronik und des Bezirksvorstandes der Kammer der Technik Frankfurt (Oder) findet in der Zeit vom 8. bis 12. Mai 1989 in Frankfurt (Oder) statt.
Um etwa 2500 Teilnehmern die Möglichkeit einer umfassenden Information über die Entwicklung, Produktion und Anwendung moderner mikroelektronischer Bauelemente zu geben, sind zwei aufeinander folgende Veranstaltungen mit gleichem Fachvortrags- und Ausstellungsprogramm vorgesehen.
Die Zielstellung dieses Symposiums besteht darin, zur weiteren beschleunigten Entwicklung und Anwendung der Mikroelektronik das hohe Leistungsniveau der Mikroelektronik unserer Republik im 40. Gründungsjahr der DDR zu dokumentieren.
Die Dokumentation der Leistungsfähigkeit der Mikroelektronik erfolgt durch etwa 40 Fachvorträge von Spezialisten der Bauelementehersteller, der Anwenderindustrie sowie aus Einrichtungen der angewandten Forschung. Ergänzt werden die Fachvorträge durch eine Dokumentation der Anwendung der vorgestellten Bauelemente in einer Ausstellung, die auf 1200 m^2 Fläche mehr als 100 Exponate vorstellt.
Neben individuellen Konsultationsmöglichkeiten bietet die Ausstellung auch die Möglichkeit von Fach- und Posterdiskussionen sowie von Gruppenführungen. Stärker als bisher wird die höhere Qualität der Zusammenarbeit zwischen Bauelementeherstellern und -anwendern verdeutlicht. Insbesondere werden die Anwenderleistungen für den Bauelemententwurf sowie der Anteil der Hoch- und Fachschulen, der Akademie der Wissenschaften und anderer wissenschaftlicher Institute an der Entwicklung der Mikroelektronik dargestellt.
Teilnahmemeldungen sind bis zum 28. Februar 1989 an folgende Adresse möglich:
VEB Halbleiterwerk Frankfurt (Oder), Betriebssektion der KDT, Postfach 379, Frankfurt (Oder), 1200 *W.*

Mikroprozessorsystem K 1810 WM 86

Hardware · Software · Applikation (Teil 5)

Prof. Dr. Bernd-Georg Münzer
(wissenschaftliche Leitung),
Dr. Günter Jorke, Eckhard Engemann,
Wolfgang Kabatzke, Frank Kamrad,
Hellfried Schumacher, Tomasz Stachowiak
Wilhelm-Pieck-Universität Rostock,
Sektion Technische Elektronik,
Wissenschaftsbereich Mikrorechentechnik/
Schaltungstechnik

- **Kommandooptionen**

LINK86 erlaubt eine Reihe von Optionen, die zur Vereinfachung der Arbeit auch als Abkürzungen angegeben werden können. Nachfolgend werden die möglichen Optionen, ihre Abkürzungen (halbfett) und ihre Wirkungen gezeigt.

Option	Wirkung
CODE	steuert Inhalt der CODE-Sektion der CMD-Datei
DATA	steuert Inhalt der DATA-Sektion der CMD-Datei
EXTRA	steuert Inhalt der EXTRA-Sektion der CMD-Datei
STACK	steuert Inhalt der STACK-Sektion der CMD-Datei
X1	steuert Inhalt der X1-Sektion der CMD-Datei
X2	steuert Inhalt der X2-Sektion der CMD-Datei
X3	steuert Inhalt der X3-Sektion der CMD-Datei
X4	steuert Inhalt der X4-Sektion der CMD-Datei
FILL	Einfügen von Nullen und nichtinitialisierten Daten in die CMD-Datei
NOFILL	*kein* Einfügen von Nullen und nichtinitialisierten Daten in die CMD-Datei
INPUT	Lesen einer Kommandozeile aus einer Eingabedatei
MAP	Erzeugen einer MAP-Datei
LIBSYMS	Einfügen von Symbolen aus Bibliotheksdateien in die SYM-Datei
NOLIBSYMS	*kein* Einfügen von Symbolen aus Bibliotheksdateien in die SYM-Datei
LOCALS	Einfügen lokaler Symbole in die SYM-Datei
NOLOCALS	*kein* Einfügen lokaler Symbole in die SYM-Datei
SEARCH	Durchsuchen einer Bibliotheksdatei und Linken der Module, auf die Bezug genommen wird

- **Dateioptionen**

Zur Beeinflussung des Inhaltes der CMD-Datei sind nachfolgend aufgeführte Optionen möglich:

Parameter	Wirkung
ABSOLUTE	absolute Ladeadresse für die Sektion der CMD-Datei
ADDITIONAL	zusätzliche Speicherzuordnung für die Sektion der CMD-Datei
CLASS	Klassen, die in eine Sektion der CMD-Datei eingeschlossen werden
GROUP	Gruppen, die in eine Sektion der CMD-Datei eingeschlossen werden
MAXIMUM	maximale Speicherzuordnung für eine Sektion der CMD-Datei
ORIGIN	Anfang des 1. Segmentes in der CMD-Datei
SEGMENT	Segmente, die in eine Sektion der CMD-Datei eingeschlossen werden

Die MAP-Option erlaubt die Angabe folgender Zusatzoptionen:

OBJMAP/NOOBJMAP
Eingabe/Nichteingabe von Segmentinformationen aus OBJ-Dateien in MAP-Dateien
L86MAP/NOL86MAP
Eingabe/Nichteingabe von Segmentinformationen aus L86-Dateien in MAP-Dateien
ALL
Alle Informationen werden in die MAP-Datei übernommen.

LINK86 hat standardmäßig folgende Optionen eingestellt:

FILL
LOCALS
NOLIBSYMS
OBJMAP
NOL86MAP

- **Optionen der E/A-Geräte**

Die Option ○ legt die Geräte der Ursprungs- und Zieldateien fest. Allgemein wird die Option in folgender Form angegeben:

○td, wobei t den Typ und
d das Gerät angibt.

Als Typ erkennt LINK86 folgende Dateitypen:

C Kommandodatei
L Bibliotheksdatei (L86)
M Listendatei (MAP)
O Objektdatei (OBJ oder L86)
S Symboldatei (SYM)

Als Gerätekennzeichen können die Buchstaben von A bis P für Disketten-/Plattenlaufwerke und die Buchstaben

X für Terminal (CON:)
Y für Drucker (LST:)
Z für Nulldevice (NULL:)

angegeben werden.

Zum Trennen mehrerer ○-Optionen bei der Angabe mehrerer ○-Zeichen müssen Kommas verwendet werden. Wird das ○-Zeichen nur einmal angegeben, sind die ○-Optionen durch Leerzeichen abzugrenzen.

- **Beispiele für die Anwendung von LINK86**

a)
B>A: LINK86
PROG[CODE[SEGMENT[CODE1, CODE2], GROUP[XYU]]]
Wirkung: Einordnung der Segmente CODE1, CODE2 und aller Segmente der Gruppe XYZ in die Code-Sektion der CMD-Datei PROG.CMD

b)
B>A: LINK86
PROG[DATA[ADD[100], MAX[1000]], CODE[ABS[40]]]
Wirkung: Die DATA-Sektion erfordert mindestens 1000H Bytes zusätzlich zu den Daten in der CMD-Datei. Die DATA-Sektion kann bis zu 10000H Bytes des Hauptspeichers verwenden. Die CODE-Sektion muß auf die *Absolutadresse* 400H geladen werden.

c)
B>A: LINK86
TESTX[NOLOCALS], TEST2[LOCALS], TEST3
Wirkung: Erzeugung einer SYM-Datei, die lokale Symbole aus TEST2.OBJ und TEST3.OBJ *aber nicht* aus TEST3.OBJ enthält.

d)
B>A: LINK86
PROG1, PROG2, MATH.L86[S]
Wirkung: Erzeugung von PROG1.CMD durch Verbindung von PROG1.OBJ, PROG2.OBJ und jener Module aus MATH.L86, auf die sich PROG1.OBJ oder PROG2.OBJ beziehen

e)
B>A: LINK86 PROGZ [○SZ, ○OD, ○LB], PROGW
B>A: LINK86 PROGZ [○SZ OD LB], PROGW
B>A: LINK86 PROGZ [○SZODLB], PROGW
Alle drei Kommandozeilen haben die gleiche Wirkung. Sie sind nur in den verschiedenen möglichen Schreibweisen dargestellt.
Wirkung: Erzeugung von PROGZ.CMD auf Laufwerk B:, Unterdrückung von

PROGZ.SYM, Lesen von PROGZ.OBJ und PROGW.OBJ von Laufwerk D: und Suchen der Bibliothek auf Laufwerk B:.

6.5 Die Debugger DDT86 und SID86

DDT86 und SID86 stellen ein aufwärtskompatibles Debugger-Set dar, wobei der DDT86 Bestandteil des SID86 ist. SID86 realisiert gegenüber DDT86 das symbolische Assemblieren und Reassemblieren. Außerdem erlaubt SID86 das Setzen von Protokollpunkten. SID86 nutzt zur symbolischen Testung die jeweilige SYM-Datei. Die Nutzung der SYM-Datei ist optional.

6.5.1 Bedienung von DDT86 und SID86

Der Debugger **DDT86** wird nach folgender Vorschrift aufgerufen und gestartet:

1) **B>A: DDT86**
   oder
2) **B>A: DDT86 <file>**.

Das erste Kommando lädt DDT86 und startet es. Nach der Ausgabe des Kommandopromptes (–) ist DDT86 arbeitsbereit. Das zweite Kommando lädt DDT86 und startet es. DDT86 lädt, nachdem es gestartet wurde, die mit <file> spezifizierte Datei. Fehlt der Dateityp, wird .CMD angenommen. Es können keine H86-Dateien geladen werden. Das zweite Kommando kann auch durch folgende Kommandofolge ersetzt werden:

B>A: DDT86
DDT86 Vx.x
–E <file>.

SID86 kann durch eines der folgenden Kommandos gestartet werden:

1) **B>A: SID86**
   oder
2) **B>A: SID86 <file>**
   oder
3) **B>A: SID86 <file> <symfile>**
   oder
4) **B>A: SID86 * <symfile>**.

Die ersten beiden Kommandoformen sind analog denen des DDT86. SID86 meldet sich nach dem Start mit dem Kommandoprompt #. Das dritte Kommando lädt die zu testende Datei und die Symboldatei. Das vierte Kommando lädt nur die Symboldatei. Die Kommandofolgen 2.), 3.) und 4.) lassen sich durch folgende ersetzen:

2) **B>A: SID86**
   SID86 Vx.x
   #E <file>
3) **B>A: SID86**
   SID86 Vx.x
   #E <file> <symfile>
4) **B>A: SID86**
   SID86 Vx.x
   #E * <symfile>.

Abkürzungen zu den Kommandos

s	20-Bit-Anfangsadresse
d	20-Bit-Zieladresse
f	16-Bit-Offset im spezifizierten Segment
b	Bytewert
w	Wortwert
bn	Unterbrechungspunkt
W	Anzeige Wortweise
S	Anzeige Segmentregister
R	Registerspezifikation
F	Flagspezifikation

6.5.2 Kommandos von DDT86 und SID86

Folgende Kommandos sind möglich:

A *Assembling*
As
Eingabe von Assembleranweisungen, s = 20-Bit-Adresse, wo die Assemblierung beginnt. Es gilt im wesentlichen die Assemblernotation nach ASM86/RASM86. DDT86 kann für s *nur* absolute hexadezimale Werte verarbeiten, während SID86 auch symbolische Ausdrücke verarbeiten kann.

Beispiele:

```
DDT86
-A1000:0
1000:0 MOV DX, 100
1000:3.
-
SID86
#A1000:0
1000:0 MOV DX, 100
1000:3
#
oder, falls 100H = WERT
#A1000:0
1000:0 MOV DX,.WERT
1000:3.
#
```

B *Blockcompare*
Bs1, f1, s2
s1 = 20-Bit-Adresse des Beginns des ersten Speicherblockes
f1 = Offset des letzten Bytes des Speicherblockes
s2 = 20-Bit-Adresse des Beginnes des 2. Speicherbereiches
Jede Differenz zwischen den Speicherbereichen wird auf dem Bildschirm angezeigt.

Beispiel:
B1000:0, 2FF, 2000:0
Vergleich von 300H Bytes ab 1000:0H mit dem Block ab 2000:0H.

D *Display*
a) **D**
b) **Ds**
c) **Ds,f**
d) **DW**
e) **DWs**
f) **DWs,f**

E *Programm laden, Symbole laden*
a) **E <file>**
b) **E <file> <symfile>** nur SID86
c) **E * <symfile>** nur SID86
d) **E**

Die Form a) lädt die durch <file> angegebene Datei. Wenn die Datei vollständig geladen ist, zeigen DDT86/SID86 die Start- und Endadresse jedes geladenen Segmentes an. Die Formen b) und c) sind bereits in 6.5.1 erläutert. Die Form e) gibt alle Segmentbereiche vorher geladener Programme wieder frei.

F *Fill*
a) **Fs, f, b**
b) **FWs, f, w**

Die Form a) speichert den 8-Bit-Wert b von s bis f. In der Form b) wird der 16-Bit-Wert w von s bis f gespeichert (in der Standardform: Low-Teil, High-Teil).

Beispiel:
F1000:0, 2FF, 55
füllt den Speicherbereich von 1000:0H bis 1000:2FFH mit 55H. Wird die Segmentadresse weggelassen, wird das aktuelle Segment angenommen.

G *Go (Programmstart)*
a) **G**
b) **G, b1**
c) **G, b1, b2**
d) **Gs**
e) **Gs, b1**
f) **Gs, b1, b2**
g) **–G** (nur SID86)

Die Formen a), b) und c) sind ohne spezifizierten Startpunkt. Ihr Startpunkt wird aus dem aktuellen CS und IP gebildet. Bei den Formen d) bis f) wird eine Startadresse mit angegeben. Die Formen b), c), e) und f) geben einen oder mehrere Unterbrechungspunkte an. Die Form g) unterdrückt die Ausgabe von Protokollpunkten (nur SID86).

L *List*
a) **L**
b) **Ls**
c) **Ls, f**

Ohne Parameter gibt das L-Kommando 12 Zeilen reassemblierten Maschinencode ab der aktuellen Adresse aus. Ist der Parameter s gesetzt, wird vor der Ausgabe die Anfangsadresse auf s gesetzt, und es werden 12 Zeilen ausgegeben. Die letzte Form reassembliert Maschinencode von s bis f und gibt ihn fortlaufend aus.

M *Move*
Ms, f, d
Das M-Kommando bewirkt den Transport eines Speicherbereiches von s bis f nach d. Falls für d kein Segment spezifiziert wurde, wird der gleiche Wert wie bei s angenommen.

R *Read*
R <file>
Das R-Kommando liest eine Datei in einen zusammenhängenden Speicherbereich (ohne Bereichsaufspaltung). Das R-Kommando gibt keine Speicherbereiche frei, die durch frühere R- oder E-Kommandos belegt wurden. Die Anzahl der zu ladenden Dateien ist auf 7 begrenzt.

S *Substitute*
a) **Ss**
b) **Sw**

Mit Hilfe des S-Kommandos kann der Inhalt von Bytes oder Worten im Speicher geändert werden. Die Speicheradressen und die alten Speicherinhalte werden nach Kommandoeingabe angezeigt. Eine Eingabe gültiger Hexeadezimalwerte überschreibt diese. Die Eingabe von nur RETURN läßt den Inhalt unverändert. Eine Anzeige bzw. Änderung ist bis zur Eingabe eines Punktes (.) oder eines unerlaubten Wertes möglich.

T *Trace*
a) **T**
b) **Tn**
c) **TS**
d) **TSn**

Das T-Kommando bewirkt eine Programmverfolgung im Tracemodus (für n = 1 – 0FFFFH) mit Angabe der Inhalte der CPU-Register. Bei den Formen a) und b) werden die Segmentregister nicht mit ausgegeben. Wird der Wert n nicht angegeben, wird nur ein Befehl ausgeführt. Zusätzlich wird der jeweils nächste Befehl in reassemblierter Form mit ausgegeben.

U *Untrace*
a) **U**
b) **Un**

c) **Us**
d) **USn**
Das U-Kommando ist mit dem T-Kommando identisch mit der Ausnahme, daß die Inhalte der CPU-Register *nur* vor der Ausführung des *ersten* Befehls angezeigt werden.

V *Value*
Das V-Kommando zeigt die aktuell durch das jeweilige Programm belegten Segmente an.

W *Write*
a) **W <file>**
b) **W <file>s, f**
Das W-Kommando schreibt den Inhalt eines zusammenhängenden Speicherbereiches zum Massenspeicher. Werden s und f nicht angegeben, übernehmen SID86/DDT86 die Werte von der letzten mit einem R-Kommando gelesenen Datei. *Ist die mit einem W-Kommando zu schreibende Datei bereits vorhanden, wird sie überschrieben!*

X *Anzeige der CPU-Register*
a) **X**
b) **XR**
c) **XF**
Das X-Kommando erlaubt die Anzeige des CPU-Statusses. Durch Spezifikation können einzelne Register (Form b)) oder Flags (Form c)) angezeigt oder geändert werden.

P *Pass-Point* (nur SID86)
a) **Pd, n**
b) **Pd**
c) **–Pd**
d) **–P**
e) **P**
Das P-Kommando setzt, löscht und zeigt Protokollierpunkte an. Die Formen a) und b) werden zum Setzen von Protokollierpunkten genutzt. Der Wert n in der Form a) gibt einen Durchlaufzählerwert an. Die Formen c) und d) werden genutzt, um Protokollierpunkte zu löschen. Die Form d) löscht alle Protokollierpunkte. Die Form e) zeigt alle aktiven Protokollierpunkte an.

7. Koprozessoren

Eine der wichtigsten Methoden zur Erhöhung der Effektivität von Mikrorechnersystemen ist die Parallelarbeit von mehreren Prozessoren in einem Rechner bei wechselseitigem Austausch von Informationen.
Informationsaustausch kann über den gemeinsamen Speicher oder E/A-Ports stattfinden. Das erste Architekturprinzip bezeichnet man als eng gekoppeltes System (tightly/closely/coupled microprocessor systems) und das zweite Prinzip als lose gekoppeltes System (loosely coupled microprocessor systems).
Die 8086-CPU besitzt einen 6-Byte-FIFO-Instruction-Queue in der Bus-Interface-Unit, welcher dem Prozessor ein vorausschauendes Befehlsholen ermöglicht. Solcher Warteschlangenmechanismus der CPU trägt in Multiprozessorsystemen zur Erhöhung des Parallelitätsgrades bei.
In diesem Beitrag sollen die spezialisierten Koprozessoren des 8086-Systems
– Arithmetikkoprozessor 8087
– Ein-/Ausgabe-Prozessor 8089
vorgestellt werden, die mit der 8086-CPU ein eng gekoppeltes Multiprozessorsystem darstellen.

Tafel 7.1 Zusammenstellung von Befehlsausführungszeiten

Gleitkomma-operation	Ausführungszeit in µs	
	8087 (5MHz)	8086 Emulator
ADD/SUBTRACT	14/18	1600
Multiply (single precision)	19	1600
Multiply (extended precision)	27	2100
Divide	39	3200
Compare	9	1300
Load (double precision)	10	1700
Store (double precision)	21	1200
Square root	36	19600
Tangent	90	13000
Exponentiation	100	17100

7.1 Systemkonfiguration mit Arithmetikkoprozessor 8087

Der Arithmetikkoprozessor 8087 ist nur in Zusammenarbeit mit der 8086/88-CPU einsetzbar. Seine interne Struktur ermöglicht die Ausführung von numerischen Operationen mit hoher Geschwindigkeit und Präzision. Tafel 7.1 zeigt eine Zusammenstellung von Befehlsausführungszeiten beim Koprozessor 8087 im Vergleich zur 8086-CPU-Emulation. Für den Anwender erscheint die Verbindung der CPU mit dem Arithmetikkoprozessor als ein komplexer Mikroprozessor mit vergrößertem Befehlsvorrat. Der 8087 liefert dem System neue Datentypen, neue Register und 68 neue Befehle. Er arbeitet parallel zur Master-CPU, das heißt, er dekodiert parallel den Befehlsstrom, führt aber nur diejenigen Befehle aus, die für ihn bestimmt sind (ESCAPE-Befehle). Die schaltungstechnische Realisierung der Verbindung von Arithmetikkoprozessor 8087 mit der 8086-CPU zeigt Bild 7.1. Die Statussignale $\overline{S0}$–$\overline{S2}$ und die Queue-Statussignale QS0–QS1 ermöglichen dem 8087 das „Mithören" und Dekodieren der Befehle parallel zur CPU.
Zum Synchronisieren wird das BUSY-Signal benutzt, das mit dem $\overline{TEST}$-Eingang der CPU verbunden ist und von der CPU abgefragt wird.
Der Koprozessor kann in einem ERROR- oder EXCEPTION-Fall die Programmbearbeitung der CPU mit einem Interrupt unterbrechen, der über den programmierbaren Interrupt-Controller 8259A an die CPU weitergeleitet wird.
Wie aus Bild 7.1 ersichtlich, werden beide Prozessoren ohne zusätzlichen Hardwareaufwand miteinander verbunden. Der Adreß-, Daten- und Steuerbus wird von beiden Prozessoren gemeinsam genutzt. Diese Konfiguration erfordert eine Arbitrierungslogik zur Kontrolle der aktuellen Buszuweisung.
Der Busanforderungs-, Busübernahme- und Busrückgabezyklus wird mit Hilfe der bidirektionalen $\overline{RQ}/\overline{GT}$-Leitung realisiert (Bild 7.2). Dieses Signal muß an $\overline{RQ}/\overline{GT0}$ oder $\overline{RQ}/\overline{GT1}$ der CPU angeschlossen werden. Der Request/Grant-Sequenz läuft in drei Etappen ab:

- Senden des ein Taktzyklus langen Request-Impulses an die CPU; Bedeutung: Der 8087 oder ein anderer Master fordert den lokalen Bus an.
- Der 8087 wartet auf den Grant-Impuls von der CPU; der 8087 startet einen Buszyklus nach Eintreffen des Grant-Impulses und übernimmt damit die Buskontrolle.
  Falls der Request-Impuls von einem anderen Master (8089) kam, leitet er den Impuls an seinen $\overline{RQ}/\overline{GT1}$-Anschluß weiter (Bild 7.1).
- der 8087 sendet den Release-Impuls an die CPU oder schaltet den von einem anderen Master auf der Linie $\overline{RQ}/\overline{GT1}$ empfangenen Release-Impuls durch, die CPU übernimmt wieder die Buskontrolle.

7.2 Busstruktur des Arithmetikkoprozessors 8087

Die Busstruktur des 8087 ist identisch mit der der 8086-CPU im Maximum-Mode (ver-

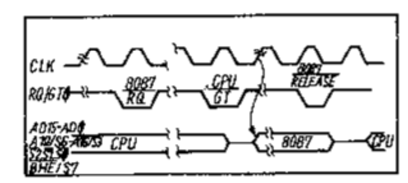


Bild 7.2 RQ/GT-Zeitdiagramm

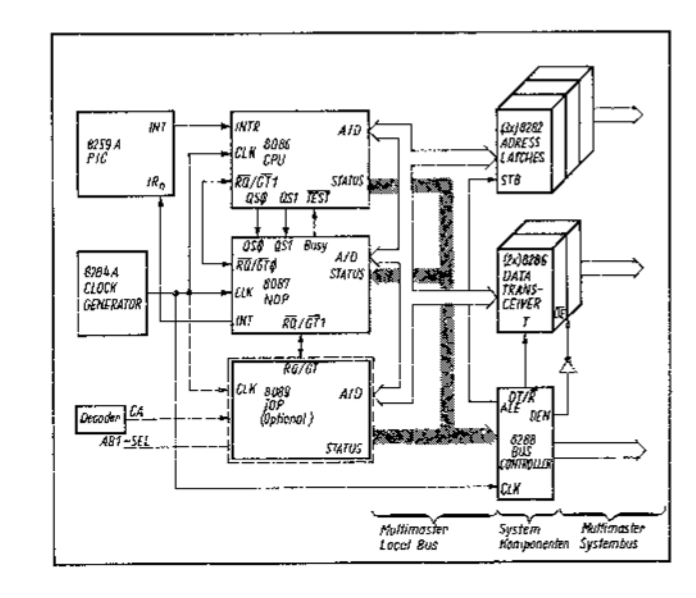


Bild 7.1 Systemkonfiguration mit Arithmetikkoprozessor 8087 und mit I/O-Prozessor 8089

gleiche Abschnitt 1.2). Der Koprozessor nutzt gemeinsam mit der CPU die Systemkomponenten: Buscontroller, Clockengenerator, Octal-Latches usw. Die Statussignale $\overline{S0}$–$\overline{S2}$ werden vom 8087 wie folgt dekodiert:

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	Bedeutung
0	X	X	nicht genutzt
1	0	0	nicht genutzt
1	0	1	Speicherlesen
1	1	0	Speicherschreiben
1	1	1	passiv

Wenn der 8087 die Buskontrolle besitzt, sind die Statussignale S6, S4 und S3 = High, während S5 = Low ist (siehe dazu Abschnitt 2.1.2). Im passiven Zustand wird vom 8087 das Statussignal S6 abgefragt, welches die Information enthält, ob die Buskontrolle von der CPU oder einem Koprozessor durchgeführt wird. S7 wird mit $\overline{BHE}$ gemultiplext und besitzt für alle 8087-Buszyklen den Wert von $\overline{BHE}$. Der 8087 enthält einen Instruction-Queue, der mit dem Queue der 8086-CPU identisch ist. Der 8087 kontrolliert die Statussignale QS0 und QS1, wodurch die Abarbeitung der sich in der Warteschlange befindenden Befehle synchron mit der 8086-CPU ablaufen kann. Diese Signale werden wie folgt dekodiert:

QS1	QS0	Bedeutung
0	0	keine Operation
0	1	erstes Byte vom Operationscode vom Queue entnommen
1	0	Queue leer
1	1	nachfolgendes Byte vom Queue entnommen

7.3 Interne Architektur des 8087

In der internen Struktur des Arithmetikkoprozessors unterscheidet man zwei Module:

- Steuereinheit (Control Unit; CU)
- Recheneinheit (Numeric Execution Unit; NEU)

Die Struktur des Arithmetikkoprozessors ist im Bild 7.3 dargestellt.

Control Unit

Die CU holt und klassifiziert die Befehle, liest bzw. schreibt die Operanden und nimmt die Synchronisation mit der CPU vor. Die 8087-Befehle befinden sich im Befehlsstrom des CPU-Programms. Die CU des Koprozessors erhält durch den Status $\overline{S0}$, $\overline{S1}$, $\overline{S2}$ und QS0, QS1 die Information, wann ein Befehlsholezyklus (Fetch) stattfindet. Die im Befehlsholezyklus gelesenen Daten (Operationscode) werden von der CU gelesen und dekodiert.

Die fünft höchsten Bits aller 8087-Operationscodes sind identisch (ESCAPE-Präfix 11011 ...) und kennzeichnen damit den Befehl als ESCAPE-Befehl. Wenn ein ESCAPE-Befehl erkannt wird, wird er von der CU dekodiert und von der CU bzw. von der NEU ausgeführt. Die Master-CPU analysiert auch alle ESCAPE-Befehle, und im Falle eines Operandentransfers vom oder zum Speicher rechnet sie die entsprechende Operandenadresse aus. Die berechnete Adresse wird in einem „leeren Lesezyklus" („dummy read") auf den Adreßbus ausgegeben und

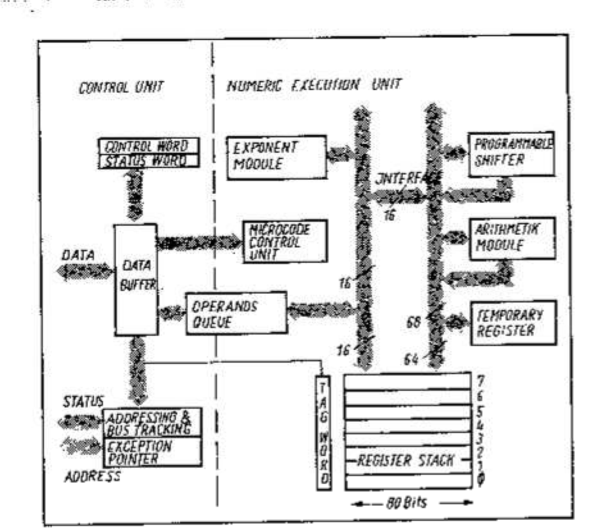


Bild 7.3 8087-Blockdiagramm

vom 8087 übernommen, wobei zwei Fälle unterschieden werden:

– *Operanden sollen gelesen werden*

Das im „leeren Lesezyklus" gelesene Datenwort wird vom 8087 übernommen. Falls der Operand länger als ein Wort ist, übernimmt der 8087 die Buskontrolle, und auf Grundlage der übernommenen Operandenadresse führt er weitere Lesezyklen durch.

– *Operanden sollen geschrieben werden*

Das im „leeren Lesezyklus" auf den Datenbus gelegte Datenwort wird ignoriert. Wenn der 8087 schreiben will, übernimmt er die Buskontrolle und führt Schreibzyklen durch, die bei der übernommenen Adresse beginnen.

Numeric Execution Unit NEU

Die NEU führt alle Befehle aus, die sich auf die Stack-Register beziehen. Dazu gehören arithmetische, logische, logarithmische, trigonometrische, Konstanten- sowie Datentransferbefehle. Der interne Datenbus der NEU ist 84 Bit breit (68 Bit Mantisse, 15 Bit Exponent, 1 Bit Vorzeichen). Während einer Befehlsabarbeitung wird das BUSY-Signal high-aktiv geschaltet und kann von der CPU abgefragt werden.

Registersatz

Der Registersatz des Arithmetikkoprozessors beinhaltet acht 80 Bit breite Stack-Register, die in folgende Felder aufgeteilt werden:

64 Bit Mantisse
15 Bit Exponent
1 Bit Vorzeichen

die vom Format dem „temporary real data type" entsprechen.

Status Word

Das Statuswort ist 16 Bit breit und beinhaltet die Informationen über den Zustand des Koprozessors. Der Status kann getestet, im Speicher mit dem Befehl FSTSW abgespeichert und vom CPU-Programm kontrolliert werden. Das Statuswort und die Bedeutung der einzelnen Bits ist in den Tafeln 7.2 und 7.3 dargestellt.

Control Word

Das Steuerwort ist 16 Bit breit, wird mit dem Befehl FLDCW geladen und dient zum Initialisieren des Koprozessors. Die Bedeutung einzelner Bits ist in Tafel 7.4 dargestellt.

Das niederwertige Byte des Steuerwortes konfiguriert Interrupt- und Exception-Verhal-

Tafel 7.2 8087 Status Word

Bit	Symbol	Bedeutung
0	IE	Invalid Operation
1	DE	Denormalized Operand
2	ZE	Zero Divide
3	OE	Overflow
4	UE	Underflow
5	PE	Precision
6	XXX	(reserviert)
7	IR	Interrupt Request
8 9 10	C0 C1 C2	siehe Tafel 7.3 siehe Tafel 7.3 siehe Tafel 7.3
11 12 13	TOP	Top of Stack Pointer 0 0 0 Register 0 TOP 0 0 1 Register 1 TOP 0 1 0 Register 2 TOP 0 1 1 Register 3 TOP 1 0 0 Register 4 TOP 1 0 1 Register 5 TOP 1 1 0 Register 6 TOP 1 1 1 Register 7 TOP
14	C3	siehe Tafel 7.3
15	B	NEU BUSY

ten des 8087. Mit Bit 7 werden generell die Interrupts gesperrt (High) oder freigegeben (Low). Mit den Bits 0–5 können die Exceptionbedingungen maskiert werden. Jede Exzeptionbedingung kann Ursache eines Interrupts sein, falls Interrupt freigegeben und Exception nicht maskiert (Low) wurde. Das Steuerwort kann mit dem Befehl FSTCW in den Speicher geladen werden.

Tag Word

Das Tag Word kennzeichnet den Inhalt der Register (Tafel 7.5) und kann mit den Befehlen FSTSW, FSAVE und FSTENV im Systemspeicher abgelegt werden.

Exception Pointers

Die Exception Pointers (Tafel 7.6) sind für die vom Programmierer geschriebenen Error-Behandlungsprogramme von Bedeutung. Während die NEU einen Befehl ausführt, speichert die CU die Operandenadresse

Tafel 7.3 Interpretation der Condition Codes

[illegible]	[illegible]	Bedeutung
[illegible]	0 X X 0	A > B
	0 X X 1	A < B
	1 X X 0	A = B
	1 X X 1	A ? B (nicht vergleichbar)
[illegible]	U 0 U U	Complete reduction
	U 1 U U	Incomplete reduction
[illegible]	0 0 0 0	Valid, positiv, unnormalized
	0 0 0 1	Invalid, positiv, exponent = 0
	0 0 1 0	Valid, negativ, exponent = 0
	0 0 1 1	Invalid, negative, exponent = 0
	0 1 0 0	Valid, positiv, normalized
	0 1 0 1	Infinity, positiv
	0 1 1 0	Valid, negativ, normalized
	0 1 1 1	Infinity, negative
	1 0 0 0	Zero, positive
	1 0 0 1	Empty
	1 0 1 0	Zero, negative
	1 0 1 1	Empty
	1 1 0 0	Invalid, positiv, exponent = 0
	1 1 0 1	Empty
	1 1 1 0	Invalid, negativ, exponent = 0
	1 1 1 1	Empty

U – Wert nach der Operation undefiniert
X – Wert wird durch die Operation nicht verändert

Tafel 7.4 Control Word

Bit	Symbol	Bedeutung	
0	IM	Invalid Operation	
1	DM	Denormalized Operand	
2	ZM	Zerodivide	Exception
3	OM	Overflow	Masks
4	UM	Underflow	1 – Exception
5	PM	Precision	is masked
6	X	(Reserved)	
7	IEM	Interrupt Enable Mask	0 Int enabled 1 Int masked
8, 9	PC	Precision Control	0 0 24 Bit 0 1 (Reserved) 1 0 53 Bit 1 1 64 Bit
10, 11	RC	Rounding Control	0 0 To Nearest or Even 0 1 Round down (toward −∞) 1 0 Round up (toward +∞) 1 1 Chop (truncate toward zero)
12	IC	Infinity Control	0 Projective 1 Affine
13, 14, 15		Reserved	

(falls vorhanden) und den Befehlscode in den Exception Pointers ab. Diese Daten können dann im Speicher abgelegt werden.

7.4 Datentypen und -formate

Der Arithmetikkoprozessor 8087 arbeitet mit sieben Datenformaten, die in drei Klassen eingeteilt werden:

- binary integers
- packed decimal integers
- binary reals

Bild 7.4 zeigt den Aufbau der 8087-Datenformate.

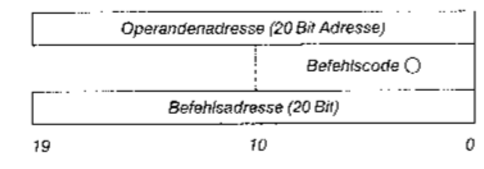
Im Bild 7.5 ist dargestellt, wie die einzelnen Datenformate im Speicher des 8086-Systems abgelegt werden.

Tafel 7.5 Tag Word

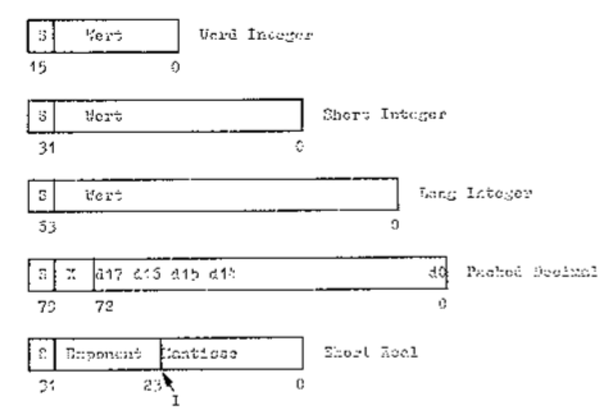
15							0
TAG(7)	TAG(6)	TAG(5)	TAG(4)	TAG(3)	TAG(2)	TAG(1)	TAG(0)

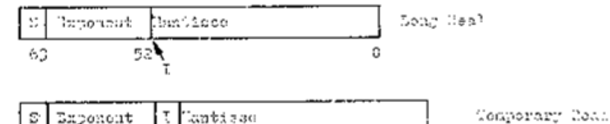
TAG-Wert: 0 0 Valid (normal or unnormal)
0 1 Zero (True)
1 0 Special (Not-a-Number, unnormal)
1 1 Empty

Tafel 7.6 Exception Pointers Format



◯ Es werden nur 11 Bits des Befehlscodes gespeichert, da die 5 höchstwertigen immer den Wert 11011B haben.





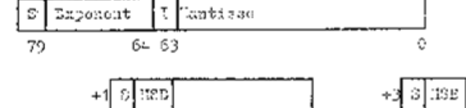


Bild 7.4 Datenformate des 8087
S – Vorzeichen (0 = positiv, 1 = negativ)
dn – Dezimalzahlen (zwei pro Byte)
X – Bits haben keine Bedeutung; werden ignoriert beim Laden; beim Abspeichern gleich Null
I – Integer-Bit der Mantisse
– Komma

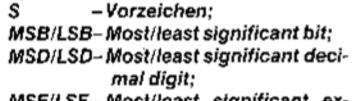


Bild 7.5 Abspeicherung der verschiedenen Datenformate im Speicher
S – Vorzeichen;
MSB/LSB – Most/least significant bit;
MSD/LSD – Most/least significant decimal digit;
MSE/LSE – Most/least significant exponent bit;
MSF/LSF – Most/least significant fraction;
I – Integer bit

Beim Laden der Daten in die internen Stack-Register und beim Abspeichern der Daten wird eine automatische Umformatierung vorgenommen.

Beispiel 1

Der Befehl FILD konvertiert die im Systemspeicher im Integer-Format abgespeicherten Daten in das 8087-interne Temporary-real-Format und lädt sie in die Stack-Register.

Beispiel 2

Der Befehl FBSTP konvertiert den Inhalt des „Top of Stack" in das Packed-decimal-Format und speichert den Wert im Systemspeicher ab.

7.5 Assemblerbefehle des 8087

Die 8087-Befehle führen Operationen im Speicher oder in internen Stack-Registern aus. Es werden folgende Gruppen von Befehlen unterschieden:

- Transferbefehle: load, store, exchange
- Arithmetikbefehle: add, subtract, multiply, divide, square root, scale usw.
- Vergleichbefehle: test, examine, compare
- Funktionen: tang, arctang, $2^x - 1$, $Y * \log_2 (X+1)$, $Y * \log_2 (x)$
- Konstanten: 0, 1, π, $\log_{10} 2$, $\log_e 2$, $\log_2 10$, $\log_2 e$
- Prozessorsteuerbefehle: FINIT, Load Control Word, Store Control Word, Enable/ Disable Interrupt, Clear Exception.

7.5.1 Datentransportbefehle

Mit Transportbefehlen können die internen 10-Byte-Daten innerhalb des Registerstapels auf die Stapelspitze umgespeichert oder bei gleichzeitiger Konvertierung in das Anwenderdatenformat vom Speicher gelesen oder in den Speicher geschrieben werden. Der Austausch mit dem Speicher erfolgt nur über die Stapelspitze. Für Speicheroperanden wird der Datentyp (Integer, Real oder Dezimal) in der mnemonischen Befehlsbeschreibung und die Datenlänge in der Operandenbeschreibung in Erweiterung der 8086-Assemblernotierung (z. B. DWORD PTR [BX]) angegeben. Der Befehl FLD führt eine Ladeoperation für Realdaten auf die Stapelspitze aus. Dabei wird der Stapelzeiger vor dem Laden auf die nächst kleinere Registernummer gestellt, so daß vorangegangene Stapeleintragungen erhalten bleiben. Ladeoperationen mit Quelloperanden im Registerstapel kopieren diese auf die Stapelspitze. Ladeoperationen aus dem Speicher sind für das 10-, 8- und 4-Byte-Format möglich. Bei 4- und 8-Byte-Realdaten erfolgt die Konvertierung auf das interne 10-Byte-Format.

Beispiele:

FLD ST(3) ; ST(0) = ST(3)
FLD ST(0) ; Duplizieren der Stapelspitze
FLD DWORD PTR [BX]; 4-Byte-Real-Format
FLD QWORD PTR [SI] ; 8-Byte-Real-Format
FLD TBYTE PTR [DI] ; 10-Byte-Real-Format

Für die Eingabe von Integer- und Dezimaldaten auf die Stapelspitze existieren Befehle mit der mnemonischen Beschreibung FILD und FBLD, die ebenfalls die Konvertierung in das interne 10-Byte-Format einschließen.

Beispiele:

FILD WORD PTR [BP] ; 2-Byte-Integer-Format
FILD DWORD PTR [SI]; 4-Byte-Integer-Format
FILD QWORD PTR [DI]; 8-Byte-Integer-Format
FBLD TBYTE PTR [BX] ; 10-Byte-Integer-Format

Die Operation FST überträgt Realdaten von der Stapelspitze in andere Register oder in den Speicher, wobei der Quelloperand auf der Stapelspitze erhalten bleibt. Die Übertragung in den Speicher ist nur für 4- und 8-Byte-Daten zugelassen. Die Verkürzung des 10-Byte-Formates unterliegt einer von vier Rundungsvorschriften, die mit dem 8087-Steuerwort eingestellt wird.

Beispiele:

FST ST(4) ; ST(4) = ST(0)
FST DWORD PTR [DI]; 4-Byte-Real-Format

Die Abspeicherung von Integer-Daten mit dem Befehl FIST ist für 2- und 4-Byte-Ergebnisdarstellungen möglich.

Beispiele:

FIST WORD PTR [SI] ; 2-Byte-Integer
FIST DWORD PTR [DI]; 4-Byte-Integer

Von größerer praktischer Bedeutung ist die Abspeicherung der Stapelspitze in Verbindung mit einer anschließenden POP-Operation, die den Registerstapelzeiger zurücksetzt. Der entsprechende Befehl FSTP für Realdaten schließt auch die Abspeicherung von 10-Byte-Daten im Speicher ein.

Beispiele:

FSTP ST(0) ; POP-Operation ohne Datentransport
FSTP TBYTE PTR [BX]; 10-Byte-Real-Daten abspeichern und POP-Operation

Die Befehle FISTP und FBSTP sind für alle Integer-Formate und das 10-Byte-Dezimal-Format gültig.

Beispiele:

FISTP QWORD PTR [SI]; 8-Byte-Integer- und ; 10-Byte-Daten
FBSTP TBYTE PTR [DI] ; abspeichern ; und POP-Operation

Der Befehl FXCH erlaubt den Datenaustausch zwischen der Stapelspitze und Stapelregistern oder Operanden im Speicher.

7.5.2 Arithmetische Grundoperationen

Die arithmetischen Operationen des 8087 enthalten die vier Grundoperationen und zwei reverse Formen für die Subtraktion und Division, bei denen die Operanden vor der Operation vertauscht werden.
In Abhängigkeit von der Form der Operandenvorgabe können die folgenden Befehlsformen unterschieden werden:

☐ Stapelabarbeitung
Die Operation bezieht sich auf die letzten beiden Stapeleintragungen ST(0) und ST(1). Das Ergebnis steht nach dem Erhöhen des Stapelzeigers um 1 an der Stapelspitze (Bild 7.6a). Die Operation wird im Assemblerprogramm ohne Operandenangabe angegeben:

FADD ; Addition im Stapel
FSUB ; Subtraktion im Stapel ; ST(1) – ST0
FMUL ; Multiplikation im Stapel
FDIV ; Division im Stapel ST(1)/ST
FSUBR; reverse Subtraktion im Stapel ; ST0 - ST(1)
FDIVR; reverse Division im Stapel ; ST0/St(1)

Bild 7.6
a) FADD
b) FADD ST(3), ST
c) FADDP ST(2), ST

☐ Registerabarbeitung
Die Operanden stehen im Register ST(0) und in einem beliebigen anderen Register. Das Ergebnis kann auf einem der Vorgaberegister abgelegt werden (Bild 7.6b).

FADD ST, ST(i)	oder	**FADD** ST(i), ST
FSUB ST, ST(i)	oder	**FSUB** ST(i), ST
FMUL ST, ST(i)	oder	**FMUL** ST(i), ST
FDIV ST, ST(i)	oder	**FDIV** ST(i), ST
FSUBR ST, ST(i)	oder	**FSUBR** ST(i), ST
FDIVR ST, ST(i)	oder	**FDIVR** ST(i), ST

☐ Registerabarbeitung mit Stapel-POP-Operation
Wenn der im Register ST(0) vorgegebene Operand nur für diese Operation benötigt wird, kann mit einer POP-Operation nach der Berechnung der in ST(0) vorgegebene Operand aus dem Registerstapel herausgeschoben werden (Bild 7.6c). Bei der Angabe des Registers ST(1) als Ergebnisregister steht das Ergebnis nach der Operation in ST(0), da für alle Register der relative Registerzeiger dekrementiert wird.

FADDP ST(i), ST ; Addition im Stapel mit POP
FSUBP ST(i), ST ; Subtraktion im Stapel mit POP
FMULP ST(i), ST ; Multiplikation im Stapel mit POP
FDIVP ST(i), ST ; Division im Stapel mit POP
FSUBRP ST(i), ST; reverse Subtraktion im Stapel mit POP
FDIVRP ST(i), ST ; reverse Division im Stapel mit POP

☐ Operation mit Speicheroperand
Die Operation bezieht sich auf die Stapelspitze und einen Operanden im Speicher im 4- oder 8-Byte-Real- oder im 2- oder 4-Byte-Integer-Format. Für die Integerformate gel-

ten die Bezeichnungen FIADD, FISUB, FIMUL, FIDIV, FISUBR und FIDIVR.

Beispiele:

FIADD WORD PTR [SI] ; Addition mit 2-Byte -Integer-Daten

FDIV DWORD PTR [BX]; Division mit 4-Byte -Real-Daten

7.5.3 Spezielle arithmetische Operationen

Die Berechnung der Quadratwurzel mit dem Befehl FSQRT überschreibt das Argument an der Stapelspitze. Die Vorgabe von Argumenten < 0 führt zur Fehlermarkierung. Die Multiplikation und Division mit ganzzahligen Potenzen von 2 kann durch Addition auf den Exponenten der internen Gleitpunktdualdarstellung einfach und schnell ausgeführt werden. Der entsprechende Befehl FSCALE benutzt den Inhalt der vorletzten Stapeleintragung ST(1) als vorzeichenbehafteten 2-Byte-Integer-Skalierungsfaktor für ST(0). Die Operation FPREM ergibt eine Modulo-Division von ST(0) mit dem Modulus in ST(1). Die Operation wird durch sukzessive bewertete Subtraktionen ausgeführt, bis der verbleibende Rest kleiner als der Modulus ist. Das Vorzeichen des Ergebnisses stimmt mit dem der Vorgabe in ST überein.

Die Rundung auf Integerwerte mit dem Befehl FRNDINT unterliegt genau wie die Konvertierung der internen Gleitpunktdarstellung in die Integerdarstellungen der mit dem 8087-Steuerwort eingestellten Rundungsvorschrift.

Der Befehl FXTRACT zerlegt den Wert an der Stapelspitze in den Wert des originalen (nicht verschobenen) Dualexponenten und den Mantissenwert. Der Mantissenwert an der neuen Stapelspitze ST(0) und der Wert des vorzeichenbehafteten absoluten Exponenten in ST(1) sind in dem internen 10-Byte-Dualdatenformat (mit Exponentenverschiebung) dargestellt.

Die Befehle FABS und FCHS bilden den Betrag und das Komplement von ST(0) durch Beeinflussung des Vorzeichenbits.

7.5.4 Vergleichsbefehle

8087-Berechnungsergebnisse können nach der Auslagerung in den Speicher mit 8086-Vergleichsbefehlen überprüft werden und Programmverzweigungen steuern.

Rechenzeitgünstiger sind Vergleichsoperationen im 8087-Registerstapel. Die Vergleichsergebnisse werden im Bedingungscodefeld des 8087-Statuswortes markiert. Nach der Abspeicherung des Statuswortes wird der Bedingungscode mit 8086-Befehlen ausgewertet.

Die Vergleichsoperation FCOM bezieht sich auf die bei den arithmetischen Grundoperationen beschriebenen Operandenformen der Stapel- und Registerabarbeitung und der 4- und 8-Byte-Realdaten im Speicher.

Beispiele:

FCOM ; Vergleich ST mit ST(1)

FCOM ST(5) ; Vergleich ST mit ST(5)

FCOM DWORD PTR [SI]; Vergleich ST mit 4-Byte-Real-Daten

Für den Vergleich mit 2- oder 4-Byte-Integer-Daten im Speicher existiert der Befehl FICOM.

Vergleichsoperationen mit anschließender POP-Operation FCOMP und FICOMP löschen den Operanden an der Stapelspitze. Für den Vergleich von ST mit ST(1) existiert zusätzlich der Befehl FCOMPP mit der anschließenden zweimaligen POP-Operation. Den Vergleich von ST mit dem Wert 0 vollzieht der Befehl FTST. Das Vergleichsergebnis für alle Vergleichsoperationen enthält das Bedingungscodefeld des Statuswortes:

C3	**C0**	
0	0	ST > 2. Operand
0	1	ST < 2. Operand
1	0	ST = 2. Operand
1	1	kein Vergleich möglich

Eine umfangreichere Statusinformation von ST ergibt der Befehl FXAM in den Bits C0, C1, C2 und C3 des Statuswortes.

7.5.5 Transzendente Funktionen

Die trigonometrischen und zyklometrischen Standardfunktionen lassen sich auf die Tangens- und Arcustangens-Funktion zurückführen. Die Berechnungsgrundlage der Funktion tan(Z) liefert der Befehl FPTAN mit einer Argumentenvorgabe $0 < Z < \frac{\pi}{4}$ in der Stapelspitze. Das Ergebnis entsteht in Form zweier Werte X (in ST) und Y (in ST(1)). Der Quotient Y/X ergibt den Tangenswert. Die Umkehrfunktion FPATAN berechnet Z = arctan (Y/X) = arctan (ST(1)/ST) mit der Bedingung $0 < Y < X < \infty$. Das Ergebnis überschreibt beide Vorgabeoperanden.

Die Exponentialfunktion zur Basis 2 kann als Grundlage der Berechnung der Exponentialfunktionen auf andere Zahlenbasen (z. B. 10, e) dienen.

Der 8087-Befehlsvorrat enthält die Operation F2XM1 für die Berechnung der Funktion $Y = 2^x - 1$ für den Wertebereich $0 \leq X \leq 0{,}5$. Das Ergebnis ersetzt die Vorgabe in ST. Für die Berechnung von Logarithmusfunktionen ist der Befehl FYL2X geeignet. Aus den Argumenten X in ST und Y in ST(1) mit den Wertebereichen $0 < X < \infty$ und $-\infty < Y < +\infty$ berechnet FYL2X die Funktion $Z = Y * \mathrm{ld}(X)$. Mit $Y = \log_g(2)$ kann damit der Logarithmus zur Basis g berechnet werden. Für höhere Genauigkeitsanforderungen kann die Funktion $Z = \mathrm{ld}(X+1)$ mit dem Befehl FYL2XP1 berechnet werden.

Die wichtigsten der für die Umrechnung aller Standardfunktionen und ihrer vollen Argumentbereiche auf die 8087-Funktionen benötigten Konstanten werden durch spezielle Ladebefehle bereitgestellt:

FLDZ ; ST(0) = 0
FLD1 ; ST(0) = 1
FLDPI ; ST(0) = π
FLDL2T ; ST(0) = ld(10)
FLDL2E ; ST(0) = ld(e)
FLDLG2; ST(0) = lg(2)
FLDLN2; ST(0) = ln(2)

7.5.6 Prozessorsteuerbefehle

8087-Steuerbefehle dienen der Initialisierung, der Programmierung der Betriebsweise und der Interruptverarbeitung.

Mit FINIT wird der 8087 in der gleichen Weise, wie nach einem RESET-Signal initialisiert. Die Freigabe und Sperre von 8087-Interrupts ist mit den Befehlen FENI und FDISI möglich. Ein im Speicher bereitgestelltes 8087-Steuerwort nach Tafel 7.4 wird mit FLDCW in das 16-Bit-Steuerregister übernommen. Dabei wird die Speicheradresse in der für 2-Byte-Integer-Daten üblichen Form angegeben.

Sowohl das Statuswort als auch das Steuerwort können mit den Befehlen FSTSW und FSTCW in den Speicher geschrieben werden. Der Befehl FCLEX löscht alle Ausnahmemarkierungen, das Interruptanforderungs- und das BUSY-Bit im Statuswort.

Für die Interruptbehandlung existieren im 8087 leistungsfähige Befehle für die blockweise Auslagerung der 8087-Informationen in den Speicher und die entsprechenden Rückladeoperationen.

Die Befehle FSTENV und FLDENV dienen dem Abspeichern und Zurückschreiben von Status-, Steuer- und Tag-Wort und der Zeiger für den zuletzt bearbeiteten Speicheroperanden und für den letzten Befehl.

Noch leistungsfähiger ist das Befehlspaar FSAVE und FRSTOR, das zusätzlich die drei letzten 10-Byte-Stapeleintragungen ST, ST(1) und ST(2) abspeichert bzw. zurückschreibt.

Weitere Steuerbefehle FINCSTP und FDECSTP übernehmen das Inkrementieren und Dekrementieren des Registerstapelzeigers. Der Befehl FFREE setzt eine 'Leer'-Markierung im Tag-Wort für ein ausgewähltes Register.

7.5.7 Synchronisation der 8086-8087-Parallelarbeit

Die in einem gemeinsamen Befehlsstrom enthaltenen 8086- und 8087-Befehle werden von beiden Prozessoren parallel ausgewertet.

Die CPU kann noch während der Bearbeitung des 8087-Befehls durch den Arithmetikprozessor, nach der BUS-Freigabe, die Bearbeitung der nächsten 8086-Befehle übernehmen.

Vor der Verarbeitung des nächsten 8087-Befehls durch beide Prozessoren muß jedoch gesichert sein, daß der Arithmetikprozessor die Abarbeitung des vorherigen Befehls beendet hat. Das gilt auch, wenn ein folgender 8086-Befehl Ergebnisse des vorangegangenen 8087-Befehls verwertet.

Durch das Einfügen von 8086-WAIT-Befehlen kann die zeitliche Synchronisation erreicht werden.

Die schaltungstechnische Grundlage dieser Zeitsteuerung bildet die Anschaltung des BUSY-Ausgangs des 8087 an den $\overline{\mathrm{TEST}}$-Eingang der CPU.

Ein WAIT-Befehl vor einem 8087-Befehl garantiert, daß die Abarbeitung dieses Befehls durch den 8086 erst beginnt, wenn der Arithmetikprozessor seine Verarbeitungsbereitschaft durch ein nichtaktives BUSY-Signal meldet. Um dieses abzusichern, erzeugen einige Assembler für die 8087-Befehle automatisch vorangestellte 8086-WAIT-Maschinenbefehle. Durch spezielle mnemonische Beschreibungen für einige 8087-Steuerbe-

fehle kann für diese die Hinzunahme des WAIT-Befehls unterdrückt werden. Zur Differenzierung von WAIT-Befehlen durch Assemblerprogramme existiert eine zusätzliche mnemonische Beschreibung FWAIT für den 8086-WAIT-Befehl. Die FWAIT-Befehle können durch eine Einstellung des Assemblerprogrammes annulliert werden (z.B. bei 8086-Emulatorprogrammen für die 8087-Befehle).

7.5.8 Beispielprogramm

Das Beispielprogramm berechnet den Logarithmus eines Argumentes zu einer beliebigen Basis. Das Argument wird in das Stackregister 1 und die Basis in den *Top of Stack* geladen. Das Ergebnis steht im *Top of Stack*.

```
;
; Übergabe:
;           TOS: Basis
;           ST1 : Argument
;           ST7 : muss frei sein
; Rückgabe:
;           TOS: Ergebnis
;       ST6, ST7 : frei
;       Die übrigen Registerinhalte
;       bleiben unverändert.

 cseg       0
log:
 call       logarithmieren
                ; Logarithmieren
                ; der Basis

; Logarithmieren des Arguments
;
 fxch       st1     ; TOS: Argument
                    ; ST1:log (basis)
 call       logarithmieren
 fdivdp     st1
 mov        byte ptr log-wahl, false
 ret
logarithmieren:
 fld1
 fxch       st1     ; TOS: Argumente, ST1: 1
 fYl2x              ; TOS: = ST1 * log2(TOS)
 ret
 dseg       0
log-wahl    db      0
false       equ     0
```

7.6 Der Input/Output-Mikroprozessor 8089

Im folgenden Abschnitt wird ein weiterer Mikroprozessor des 8086-Systems vorgestellt, der durch seine Multiprocessing-Mechanismen, DMA-Eigenschaften und einen auf Peripheriebedingungen spezialisierten Befehlssatz zur Steigerung der Leistungsfähigkeit des Systems beiträgt.

7.6.1 Entwicklung der Peripheriebaugruppen der Mikroprozessorsysteme (Bild 7.7)

Die erste Generation der Mikroprozessoren war durch mit TTL-Schaltkreisen aufgebaute Peripheriesteuerungen gekennzeichnet. In der nächsten Generation wurden Single-Chip Peripherie-Controller eingeführt, die durch Programmierung in der Anwendung komfortabler waren. Der Datentransfer von bzw. zur Peripherie wurde weiterhin von der CPU kontrolliert. Mit der Einführung von DMA-Schaltkreisen (Direct Memory Access) konnten schnelle Datentransferoperationen auch unabhängig von der CPU realisiert werden. In der weiteren Entwicklung wurde der DMA-Schaltkreis mit Eigenschaften eines Mikroprozessors versehen und zu einem spezialisierten Input-/Output-Prozessor vereinigt. Dieser bildet mit der zentralen CPU ein Multiprozessorsystem und entlastet durch Parallelarbeit die CPU von zeitaufwendiger Peripheriebedienung. Ein Input-/Output-Prozessor, der auf einem Chip die Eigenschaften eines DMA und Prozessors integriert, ist der Input-/Output-Prozessor (IOP) 8089.

7.6.2 Der I/O-Prozessor im 8086-System

Der I/O-Prozessor 8089 bildet mit der 8086/88-CPU ein leistungsfähiges Multiprozessorsystem. Bild 7.1 zeigt eine zentrale Verarbeitungseinheit mit den Prozessoren 8086, 8087 und 8089 ohne zusätzlichen Hardwareaufwand. Der 8086 arbeitet in dieser Konfiguration als Master und die anderen Prozessoren im „local"-mode als Slave. In einer anderen Systemkonfiguration (Bild 7.8) arbeiten zwei IOPs ohne 8086-CPU im sogenannten „remote"-mode, wobei einer als Master und der andere als Slave fungiert.

Die Statussignale $\overline{S0} \ldots \overline{S2}$ sind mit dem Bus-Controller 8288 verbunden und ergeben folgende Kodierung:

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	Bedeutung
0	0	0	Instruction fetch from I/O-Space
0	0	1	Data fetch from I/O-space
0	1	0	Data store to I/O-Space
0	1	1	not used
1	0	0	Instruction fetch from system space
1	0	1	Data fetch from system space
1	1	0	Data store to system space
1	1	1	no bus cycle run

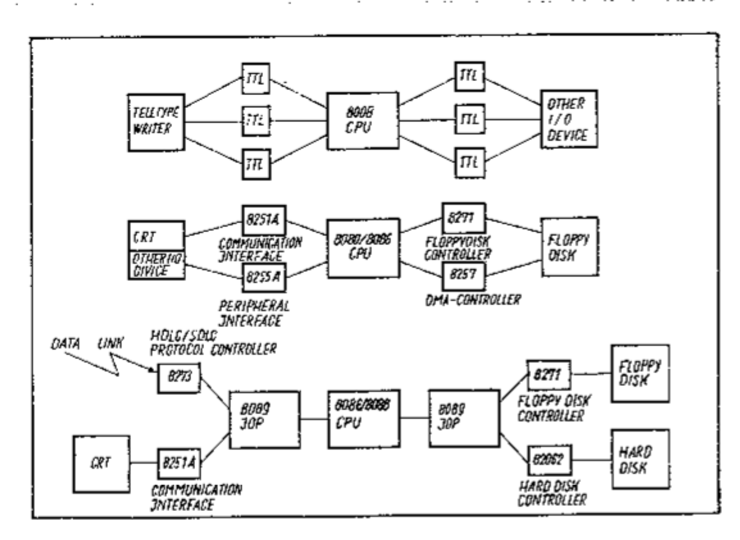
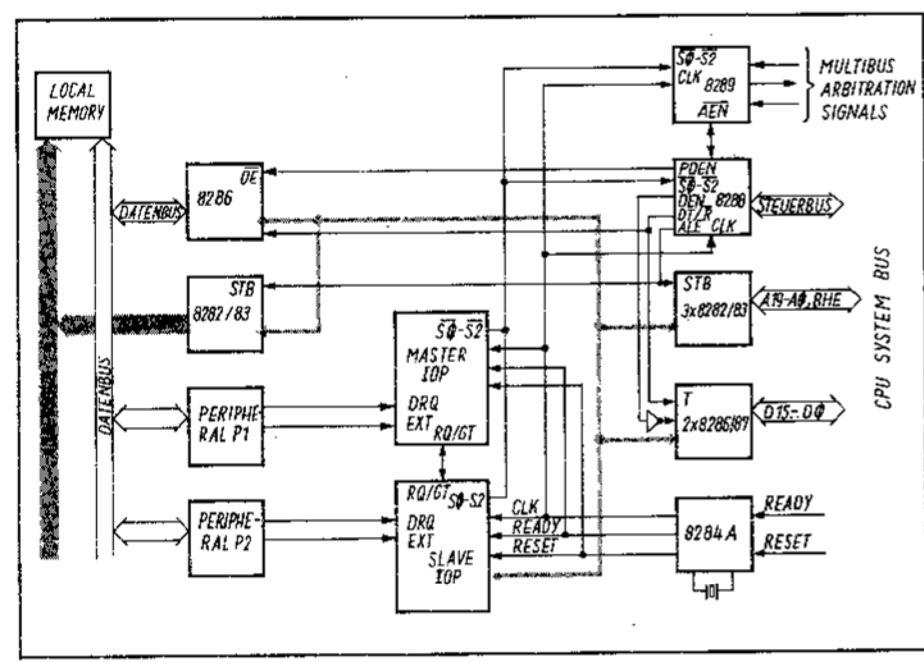


Bild 7.7 Entwicklung der Peripheriebaugruppen

Bild 7.8 Zweiprozessorsystem mit 8089 im „remote" Mode



Literatur

/1/ Dokumentation SCP 1700. VEB Robotron Elektronik Dresden

/2/ CP/M-86 Programmer's Guide. Digital Research 1981

/3/ CP/M-86 User's Guide. Digital Research 1981

/4/ Heckel, U.: Das Betriebssystem SCP 1700 EDV-Aspekte 6 (1987) 1, S. 18

wird fortgesetzt

ATAN 2-BASIC-Routinen zur Umwandlung kartesischer Koordinaten in Polarkoordinaten

Dr. Johann Komusiewicz,
Friedrich-Schiller-Universität Jena

Seit einiger Zeit läuft in Jena unter Leitung von Dr. M. Steinbach eine Vortragsreihe „6510-Prozessor" der URANIA, die sich regen Interesses erfreut.
In einer der Veranstaltungen gab Dr. Steinbach die Anregung zur Erstellung eines möglichst kurzen BASIC-Programms zur Berechnung des Polwinkels A. Die hier vorgestellten Routinen sind das Ergebnis dieses kleinen Wettbewerbs.

1. Problemstellung

Bei der Umwandlung kartesischer Koordinaten in Polarkoordinaten ist die Berechnung des Polwinkels A erforderlich (siehe Bild 1).
In FORTRAN 77 gibt es dazu die Funktion ATAN 2, wobei für A die Einschränkung $-\pi A \leqq \pi$ gilt und die Eingabe von I = R = 0 verboten ist. Ein entsprechendes BASIC-Programm ist z. B. in /1/ zu finden.
Für viele Anwendungen ist es jedoch wünschenswert, Werte von A zwischen 0 und 2π zu erhalten. PAULIN liefert dazu in /2/ ein ALGOL-Programm, das sich leicht in BASIC übertragen läßt. Ein BASIC-Programm in /3/ leistet Gleiches. Bei beiden Programmen ist jedoch die gleichzeitige Eingabe von I = R = 0 verboten. Das in /4/ angegebene Programm läßt diese Eingabe zu, arbeitet ansonsten aber nur im 1. Quadranten fehlerfrei. Die Vorgabe für den Wettbewerb – hier als Variante 0 bezeichnet – schließt diese Lücke, beansprucht mit 8 BASIC-Zeilen aber sehr viel Speicherplatz (Bild 2).

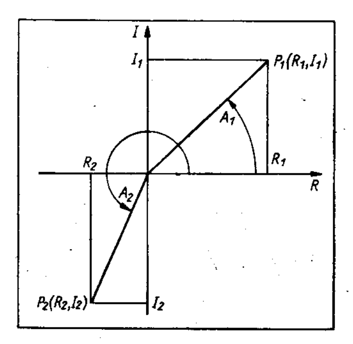


Bild 1

2. Ergebnisse

Im Wettbewerb reichten 8 Kollegen insgesamt 18 Lösungen ein. 2 Varianten mußten auf Grund von Funktionsfehlern zurückgewiesen werden.
Um für alle Programme einheitliche Testbedingungen zu sichern, wurden stets Zeilennummern ab 1000 verwendet und der Winkel im Bogenmaß berechnet. Alle Leerzeichen wurden eliminiert. Die Auswertung erfolgte nach 4 Gesichtspunkten:
(i) Korrektheit:
Werden allen zulässigen Eingabedaten die richtigen Funktionswerte zugewiesen?
(ii) Portabilität:
Ist das Programm auch auf Kleinrechnern mit anderen BASIC-Versionen lauffähig?
(iii) Speicherbedarf:
Wieviel Byte werden zur Abspeicherung der Routine benötigt?
(iv) Geschwindigkeit:
Wie groß ist der mittlere Zeitbedarf für einen Zugriff?
Alle Programme wurden auf C64 getestet; Vergleichsrechnungen mit einem KC 83/3 ergaben ähnliche Resultate. Die Ergebnisse sind in Tafel 1 zusammengefaßt.

Tafel 1

Variante	Korrektheit	Portabilität	Speicherbedarf in Byte	mittlere Rechenzeit in ms
0	+	+	127	64
1	(–)	+	86	65
2	(–)	+	95	72
3	+	+	95	82
4	(–)	+	58	86
5	(–)	+	62	102
6	+	–	73	133
7	+	(–)	77	124
8	+	+	76	77
9	+	(+)	66	68
10	+	+	70	80
11	+	(+)	95	93
12	+	(+)	67	76
13	+	+	76	65
14	(–)	+	85	75
15	+	+	98	67
16	+	+	83	87

Auf den ersten Blick fällt auf, daß die ursprüngliche Variante die kürzeste Rechenzeit aufweist. Hier zeigt sich eindrucksvoll der Vorteil der „Teile-und-herrsche"-Strategie. Andererseits benötigt z. B. Variante 4 mit 58 Byte weniger als die Hälfte des Speicherplatzes.
Die Varianten 4, 5, 8, 10 und 16 arbeiten mit logischen Variablen. Als Beispiele sind hierfür die Varianten 4 und 10 aufgeführt.
Bei den Varianten 2, 3, 6, 7, 9 und 11–15 werden die notwendigen Fallunterscheidungen über die Signum-Funktion realisiert. Hier sollen die Varianten 9 und 12 als Illustration dienen. Nun zu den einzelnen Punkten:
(i): Die Varianten 1, 14 und 2 geben für R = I = 0 die Werte A = π bzw. A = π/4 aus. In den Varianten 1, 4 und 5 tritt z. B. für R = 0, I = 100 Overflow ein.
(ii): Variante 6 nutzt die Besonderheit des C-64-BASIC, daß 0↑ – 1 keinen Fehler hervorruft, sondern 0 liefert. Die Varianten 4, 8, 10 und 16 setzen voraus, daß die logischen Variablen den Wahrheitswert –1 besitzen, was für die meisten Kleinrechner erfüllt ist.
Die Varianten 6, 7, 9, 11 und 12 sind in vielen BASIC-Versionen zu lang für eine Zeile.
(iii): Hier wurden ansprechende Resultate erreicht. Den Varianten 4 bis 10, 12 und 16 genügt eine BASIC-Zeile; auch 1, 13 und 14 benötigen nur unwesentlich mehr Speicherplatz.
(iv): Die Varianten 1, 9, 13 und 15 kommen schon nahe an die Rechenzeit von Variante 0 heran.
Nach Abwägung aller Kriterien wurde Variante 9 von Dr. A. Müller als beste Lösung ermittelt.

3. Schlußbemerkungen

Bei der Auswertung brachte mich ausgerechnet eine der fehlerhaften Varianten auf die Idee, den Komplementwinkel $\bar{A}$ = ATN(R/I) zu verwenden. Die somit erhaltenen Lösungen sind korrekt, portabel und haben bei hoher Geschwindigkeit noch geringeren Speicherbedarf, wie Tafel 2 zeigt.

Tafel 2

Variante	Korrektheit	Portabilität	Speicherbedarf in Byte	mittlere Rechenzeit in ms
17	+	+	59	61
18	+	+	50	66
19	+	+	54	62
20	+	+	45	64
21	+	+	46	66

Gegenüber der Vorlage wurde der Speicherbedarf auf die Hälfte, bzw. bei Variante 20 und 21 sogar auf etwa ein Drittel reduziert, wobei die Varianten 17 und 19 sogar einen Geschwindigkeitsgewinn erbringen.
Damit zeigte dieser Wettbewerb, daß auch „bewährte" Standardprogramme durchaus verbesserungsfähig sind.

Bild 2

```
999 REM VARIANTE 0
1000 IFR<>0GOTO1005
1001 IFI=0THENA=0
1002 IFI>0THENA=π/2
1003 IFI<0THENA=3*π/2
1004 GOTO1009
1005 A=ATN(I/R)
1006 IFR<0THENA=A+π
1007 IFR>0THENIFI<0THENA=A+2*π

999 REM VARIANTE 4
1000 A=ATN(I/(R-(R=0)/1E37))-π*(R<0)+2*π
*(I<0)*(R>=0)

999 REM VARIANTE 10
1000 A=-π*(3*(I<0)+(I>0))/2:IFRTHENA=ATN
(I/R)+π*(2*(R>0)+(I<0)-(R<0))

999 REM VARIANTE 9
1000 A=SGN(I)*(SGN(I)-.5)*π:IFRTHENA=ATN
(I/R)+π:IFR>.THENA=A-π*SGN(SGN(I)+.5)

999 REM VARIANTE 12
1000 A=SGN(I)*(SGN(I)-.5)*π:IFRTHENA=ATN
(I/R)+π-π*SGN(SGN(R)+1)*SGN(SGN(I)+.5)

999 REM VARIANTE 17
1000 IFI=.THENA=π-π*SGN(1+SGN(R))
1001 IFITHENA=π-π/2*SGN(I)-ATN(R/I)

999 REM VARIANTE 18
1000 A=π-π*SGN(1+SGN(R)):IFITHENA=π-π/2*
SGN(I)-ATN(R/I)

999 REM VARIANTE 19
1000 IFI=.THENA=-(R<0)*π
1001 IFITHENA=π-π/2*SGN(I)-ATN(R/I)

999 REM VARIANTE 20
1000 A=-(R<0)*π:IFITHENA=π-π/2*SGN(I)-AT
N(R/I)

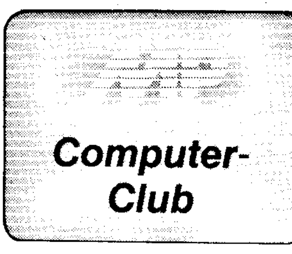
999 REM VARIANTE 21
1000 A=-(R<0)*π:IFITHENA=π/2-(I<0)*π-ATN
(R/I)
```

Literatur

/1/ Bretz, J.: 100 BASIC-Programme. Markt und Technik, München 1985
/2/ Paulin, G.: Grundlagen der Programmiertechnik. VEB Verlag Technik, Berlin 1974
/3/ Ruckdeschel, F. R.: BASIC scientific subroutines, Vol. 1. McGraw Hill, New York 1981
/4/ Poole, L.; Borchers, M.: 77 BASIC-Programme. TeWi München 1980

✉ KONTAKT ☎

Friedrich-Schiller-Universität Jena, Sektion Mathematik, Schillerstraße UHH, 17. OG, Jena, 6900, Tel. 8 22 45 70



REDABAS-Tips

Mathematische Funktionen

REDABAS realisiert die Grundrechenarten, was für die Mehrzahl der Anwendungsfälle ausreichend ist.
Durch Einbindung von Assemblerroutinen kann man die REDABAS-Arithmetik aufbessern.
Eine einfachere Möglichkeit ist die Umsetzung von Näherungsverfahren in REDABAS-(Unter)-Programme, die dann je nach Bedarf eingesetzt werden können.

Wurzelfunktion

Zur Berechnung der n-ten Wurzel einer Zahl a wird das folgende Näherungsverfahren benutzt.

$x = a^{1/n}$

a) Schätzung eines Näherungswertes x_0

b) $$\frac{a}{\underbrace{x_0^* x_0^* \ldots}_{n-1}} = a_0$$

Die Zahl a wird (n − 1)mal durch den Anfangswert x_0 dividiert.

c) $a_0 + (n-1)\,x_0 = b_0$
Addition von (n − 1)mal Anfangswert + a_0

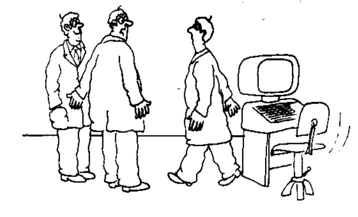
d) $b_0/n = x_1$
verbesserter Näherungswert x_1

Mit dem verbesserten Näherungswert werden die Gleichungen b bis d erneut durchlaufen, bis die Zahl x die gewünschte Genauigkeit erreicht hat.
Für die Berechnung der Quadratwurzel wurde das Näherungsverfahren in das Programm in Bild 1 umgesetzt. Gegebenenfalls müssen die Schätzung des Näherungswertes und die Wahl des Abbruchkriteriums auf die Anwendungsbedingungen der Wurzelberechnung angepaßt werden.

```
*---------------------------------------------------------------------
* WURZEL.PRG / 11.09.87 / REALISIERUNG DER QUADRATWURZEL-FUNKTION
*---------------------------------------------------------------------
* DAS BERECHNUNGSVERFAHREN IST IN DER BEDIENUNGSANLEITUNG DES
* MINIREX 75,VEB ROEHRENWERK MUEHLHAUSEN ZU FINDEN.
* DIE GENAUIGKEIT WIRD DURCH MI2 VORGEGEBEN.
*---------------------------------------------------------------------
* EINGANG: ME1 = < RADIANT  (NUMERISCH) >
*
* INTERN : MI1 = ABWEICHUNG
*          MI2 = ABBRUCHKRITERIUM
*
* AUSGANG: MA1 = < ERGEBNIS >, MA1 = (QUADRAT)WURZEL AUS ME1
*---------------------------------------------------------------------
STORE 0.5 TO MA1
STORE 0.00001 TO MI2
IF ME1 > 1
 STORE 5 TO MA1
ENDIF
IF ME1 > 50
 STORE 10 TO MA1
 STORE 0.0001 TO MI2
ENDIF
IF ME1 > 500
 STORE 20 TO MA1
ENDIF
IF ME1 > 1000
 STORE 65 TO MA1
ENDIF
IF ME1 > 10000
 STORE 200 TO MA1
 STORE 0.001 TO MI2
ENDIF
IF ME1 > 100000
 STORE 650 TO MA1
ENDIF
IF ME1 > 1000000
 STORE 2000 TO MA1
 STORE 0.01 TO MI2
ENDIF
STORE MA1*MA1-ME1 TO MI1
IF MI1 < 0
 STORE MI1 * -1 TO MI1
ENDIF
DO WHILE MI1 > MI2
 STORE (ME1/MA1+MA1)/2 TO MA1
 STORE MA1*MA1-ME1 TO MI1
 IF MI1 < 0
  STORE MI1 * -1 TO MI1
 ENDIF
ENDDO
RELEASE MI1,MI2,ME1
RETURN
*---------------------------------------------------------------------
```

Wir sprachen gerade vom intelligenten Assistenten, nicht von Ihnen!

Zeichnung: Steger



Hinweis zum A 7150

Wird am A 7150 nach Einschalten bzw. RESET der Confidence-Test nach dem PIC-Test abgebrochen und anschließend das Betriebssystem DCP gebootet, so kann es geschehen, daß bei der Arbeit nicht der komplette Hauptspeicher zur Verfügung steht. Insbesondere bei eingerichtetem virtuellem Diskettenlaufwerk kann das zu Problemen führen. Erkennbar ist dieser Zustand mit Hilfe der System-Info-Funktion von MDISK an einer Differenz von etwa 150 KByte zwischen den Angaben „Total memory reported by DCP" und „MDISK has found the total memory to be".
Als Ausweg sollte man in jedem Fall den Confidence-Test vollständig durchlaufen lassen und bei notwendigem Neuladen des Systems während der Arbeit ein RESET nur ausführen, wenn die Funktionen „Warmstart" (CTRL-ALT-DEL) bzw. „Kaltstart" (CTRL-ALT-CE) nicht mehr möglich sind:

Hinweis:
Das im Beitrag „Änderungen am Betriebssystem SCP 1700 des AC A 7100" (MP 6/88, S. 181) versehentlich unterschlagene Bild 5 soll bei dieser Gelegenheit nachgereicht werden.

Frank Isekeit, Brandenburg

```
;
;       INITIALISATION OF ALL UNITS
;
LL8:    MOV     BX,OFFSET SIGNON3   ;Aufforderung Diskettenwechsel
        CALL    PMSG                ;ausgeben
        CALL    CONIN               ;auf Quittung warten
        XOR     CX,CX                     ;UNIT Ø
        MOV     CH,NUNIT
INI1:   CMP     CL,4                      ;ELDISK ?
        JNE     INI1Ø                     ;IF NE NO
```

Autoprogrammstart mit EPROM-Modul für KC 85

Bei vielen Einsatzfällen der Kleincomputer KC 85/3 bzw. /2 werden stets die gleichen Programme benötigt. Um das Programmladen von der Kassette zu vermeiden, ist es möglich, Programme auf EPROM-Modulen abzulegen. Bei Programmen bis 8 KByte Länge ist der Modul M025 USER PROM geeignet.
Trotzdem sind nach dem Einschalten sich stets wiederholende Tastatureingaben notwendig (z. B. SWITCH, BASIC, Programmname usw.). Es gibt hierfür jedoch eine andere Möglichkeit. Bekanntermaßen schaltet das Betriebssystem CAOS beim Einschalten des Computers einen Modul M022 EXPANDER RAM im Modulschacht 8 automatisch auf der Basisadresse 4000H aktiv. Die Analyse des Betriebssystems zeigt, daß etwas ähnliches im CAOS auch für EPROM-Module vorgesehen ist. Dabei wird ein Modul mit dem Strukturbyte 01, im Modulschacht 8 beim Einschalten oder Tastenreset automatisch auf der Basisadresse 4000H aktiv geschaltet, und nach vollständiger Systeminitialisierung erfolgt ein Programmsprung zur Adresse 4000H. Die Wirkung ist die gleiche, als würde nach dem Einschalten ein Maschinenprogramm aus dem Menü aufgerufen. Es bietet sich da natürlich an, einfach ein EPROM-Modul auf das Strukturbyte 01 umzubauen. Durch diesen Autoprogrammstart ergeben sich z. B. folgende neue Einsatzfälle für Kleincomputer:
– autonom arbeitende Steuerungen (auch ohne Tastatur), die nach RESET bzw. Spannungsausfall selbständig wieder anlaufen
– automatische Initialisierung einer anderen (Profi-)Tastatur
– automatischer Programmstart mit Laden von Daten/Programmen z. B. über V.24-Modul.
Geeignet für den Autostart sind, neben Eigenbau-EPROM- bzw. CMOS-RAM-Modulen, umgebaute Module M025 USER PROM.

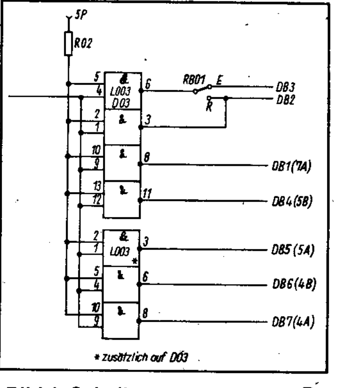


***Bild 1** Schaltungsauszug zur Erzeugung des Strukturbytes 01*

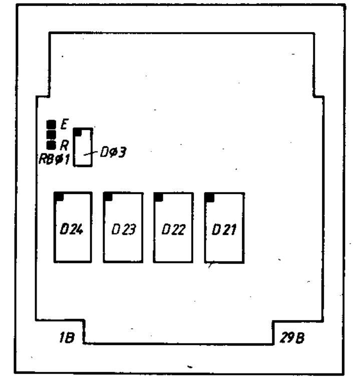


***Bild 2** Lage des Schaltkreises D03*

Umbau des Moduls M025 auf Autostart

Der Umbau des Moduls ist relativ einfach. Es muß nur das Strukturbyte von F7 auf 01 geändert werden. Bild 1 zeigt den Schaltungsauszug aus dem Gesamtstromlaufplan des Moduls, der zur Erzeugung des Strukturbytes dient. In der Originalschaltung zieht das oberste Gatter des Schaltkreises D03 (DL003) beim Eingabebefehl IN nn80H (nn-Modulschachtadresse) das Datenbit 3 auf Low. Bild 2 zeigt im Bestückungsplan zum Modul die Lage des Schaltkreises D03. Zur Erzeugung des Strukturbytes 01 müssen zusätzlich die Datenbits 1, 2, 4, 5, 6, 7 auf Low gezogen werden.

Tafel 1 Startroutine für BASIC-Autostart mit Modul

```
4000 11 00 03       LD    DE,300H     ;BASIC-Arbeitszellen
4003 21 1D 40       LD    HL,END      ;COPY-Bereich
4006 01 00 20       LD    BC,2000H    ;Länge
4009 ED B0          LDIR              ;Umladen
400B 21 18 40       LD    HL,4018H    ;Tastenfolge
400E 22 D1 B7       LD    (0B7D1H),HL ;Zeiger
4011 DD CB 08 F6    SET   6,(IX+8)    ;aktivieren
4015 C3 00 C0       JP    0C002H      ;REBASIC
4018 52 55 4E       DEFM  'RUN'
401B 0D             DEFB  0DH         ;ENTER
401C 00             DEFB  0           ;Ende
401D           END  EQU   $
```

Tafel 2 Startroutine für Kassettenaufzeichnung

```
0200 21 0D 02       LD    HL,20DH
0203 22 D1 B7       LD    (0B7D1H),HL
0206 DD CB 08 F6    SET   6,(IX+8)
020A C3 02 0C       JP    0C002H
020D 52 55 4E       DEFM  'RUN'
0210 0D             DEFB  0DH
0211 00             DEFB  0
```

Für die Bits 1, 2 und 4 werden die drei noch freien Gatter des D03 verwandt. Dazu werden die Eingänge entsprechend Bild 1 mit den bereits verschalteten Pins 4 und 5 verbunden. Der Ausgang 3 kann an den Anschluß E der Programmierbrücke RB01 angeschlossen werden. Die weiteren Ausgänge verbindet man über isolierte Drähte mit den entsprechenden Datenbits zweckmäßigerweise in der Nähe des direkten Steckverbinders. Für die Bits 5, 6 und 7 wird ein zusätzlicher Schaltkreis DL003 benutzt. Dieser kann direkt auf den D03 aufgesetzt werden. Dabei müssen nur die Ausgangspins 3, 6, 8 und 11 hochgebogen und über Drähte (nur 3, 6, und 8) angeschlossen werden. Die restlichen Pins werden mit den Pins gleicher Nummer des D03 verlötet. Nach dem Umbau kann eine Überprüfung mit dem SWITCH-Kommando erfolgen, aber natürlich nicht im Schacht 8. Es muß dabei das Strukturbyte 01 angezeigt werden.

BASIC-Autostart

Da die meisten Programme der Kleincomputer KC 85 in BASIC geschrieben sind, stellt sich natürlich die Frage, wie diese Programme von EPROM-Modulen gestartet werden können. Aufgrund der Speicherstruktur des BASIC-Interpreters /1/ ist es nicht möglich, das BASIC-Programm direkt auf dem EPROM-Modul abzuarbeiten. Deshalb ist es erforderlich, beim Start des Moduls das BASIC-Programm umzuladen. Das Programm wird zusammen mit den Arbeitszellen des Interpreters (Speicherbereich 300H bis 400H) im Modul abgespeichert. Dadurch wird es möglich, den Interpreter mit Warmstart (entspricht REBASIC) zu starten. Tafel 1 zeigt das zum Programmstart notwendige Maschinenprogramm. Das Ende des BASIC-Programmes findet man auf den Adressen 3D7H/3D8H. Die Länge beim Umladen (Tafel 1, Adresse 4007/8H) ergibt sich aus der Endeadresse minus 300H. Der Start erfolgt, wie gewohnt, mit **RUN**. Die Eingabe der RUN-Anweisung wird nach /2/ über die Funktionstastensimulation durchgeführt. Dabei ist die Programmlänge mit modifiziertem Modul M025 auf knapp 8 KByte beschränkt.
Eine Möglichkeit der EPROM-Programmierung ist in /3/ beschrieben.
Das hier beschriebene Verfahren zum BASIC-Programmstart kann auch für den Selbststart von der Kassette benutzt werden. Dazu wird der BASIC-Interpreter mit **BYE** verlassen. Ab Adresse 200H wird das Maschinenprogramm aus Tafel 2 eingegeben. Danach erfolgt das Abspeichern mit dem SAVE-Kommando:
Anfangsadresse – 200
Endeadresse – Programmende aus Zellen 3D7/3D8H
Startadresse – 200
Das Programmladen und -starten erfolgt gleich aus dem CAOS-Menü mit **LOAD**.
Weitere Möglichkeiten ergeben sich, wenn Druckertreiber oder Zeichenbildtabellen im Modul untergebracht werden.

Bernd Schübler

Literatur

/1/ Schiwon, K.; Kollmeyer, S.: BASIC-Interpreter für KC 85/2 und KC 85/3. Mikroprozessortechnik 1 (1987) 3, S. 91
/2/ Kirves, K.-D.: Arbeit mit BASIC-Datenfeldern beim KC 85/3. Mikroprozessortechnik 1 (1987) 3, S. 94
/3/ Möckel, F.: EPROM-Programmierer für KC 85/2 und KC 85/3. Radio, Ferns., Elektron. 36 (1987) 7, S. 419

Ständige Zeitanzeige auf dem KC 85/1

Das vorgestellte Programm realisiert eine ständig anzeigende Zeit für den KC 85/1. Es nutzt die Interruptmöglichkeit des CTC sowie Routinen des Betriebssystems. Das Programm wird ab Adresse XXD, vorzugsweise ab Adresse 20CH (524D) geschrieben und mit DOKE 514, -5088:DOKE 518, XXD gestartet. Der erste DOKE-Befehl legt den Ort der Zeitausgabe auf dem Bildschirm fest. Der zweite DOKE-Befehl bewirkt den Austausch der Startadresse des Interruptbehandlungsprogrammes für CTC-Kanal 3 im LOW-Byte des Interruptvektors (Zelle 206H = 518D). Das ursprünglich gerufene Interruptbehandlungsprogramm INUHR (UP Sekundentakt für Systemuhr, ab F4A8H) wird innerhalb des neuen Programms mit CALL F4A8H zusätzlich gerufen. Die Uhr wird mit POKE 29,StdD:POKE 30,MinD:POKE 31,SekD oder mit dem Time-Befehl des Betriebssystems gestellt. Das Ausschalten muß vor jedem CSAVE/CLOAD bzw. vor jeder Benutzung des CTC mit DOKE 518,-830 erfolgen.

Sven Rabe

```
20C PUSH HL       E5
20D PUSH DE       D5
20E PUSH BC       C5
20F PUSH AF       F5
210 DI            F3
211 LD DE, (0202) ED 5B 02 02
215 CALL F4A8     CD A8 F4
218 POP AF        F1
219 POP BC        C1
21A POP DE        D1
21B POP HL        E1
21C EI            FB
21D JMP FCC2      C3 C2 FC
```


Literatur

/1/ Dokumentation des Betriebssystems OS KC 85/1, VEB Robotron Vertrieb Berlin

Automatisches Einlesen der Fehlermeldungen bei TURBO-PASCAL auf dem AC 7100

Beim Start von TURBO-PASCAL wird der Anwender zuerst gefragt, ob die Fehlermeldungen nachgeladen werden sollen. Meistens wird diese Frage mit ja beantwortet, da Fehlernummern ohne erklärenden Kommentar wenig aufschlußreich sind. Demzufolge ist diese Frage überflüssig und könnte auch automatisiert werden. Dies ist mit einer kleinen Änderung in der Datei TURBO.CMD möglich, welche mit dem Debugger SID86 bequem ausgeführt werden kann. Den Debugger SID86 findet man auf der Systemdiskette für SCP 1700. Die Arbeitsgänge sind der Tafel zu entnehmen.
Beim erneuten Start von TURBO-PASCAL mit T<ENTER> werden die Fehlermeldungen automatisch nachgeladen.

Hartmut Schreiber

ANZEIGE	EINGABE	ERLÄUTERUNGEN
A>	SID86 <ENTER>	Debugger starten
#	rTURBO.CMD <ENTER>	Datei TURBO.CMD einlesen
#	a2A0A <ENTER>	auf Adresse 2A0A assemblieren
..:2A0A	mov ax,0b <ENTER>	neuen Befehl eingeben
..:2A0D	. <ENTER>	Assemblieren abbrechen
#	a2A10 <ENTER>	auf Adresse 2A10 assemblieren
..:2A10	nop <ENTER>	2 mal nop einfügen
..:2A11	nop <ENTER>	
..:2A12	. <ENTER>	
#	wT.CMD <ENTER>	neue Datei T.CMD zur Diskette
#	^C	SID86 beenden

Sichern von PC-Daten

Seit Mitte des Jahres wird von der Handelsgesellschaft Transcommerz – unseren Lesern aus Berichten über die Leipziger Messen bekannt – ein achtseitiges Informationsblatt, die TC – Transcommerz Informationen, herausgegeben. Mit freundlicher Genehmigung veröffentlichen wir aus TC 2/88 nachfolgend auszugsweise Tips zur Datensicherung.

Paßwort-Schutz

Mit entsprechenden Zusatzeinrichtungen prüft Ihr PC vor jedem Zugriff auf Daten (Verändern, Programmanwendungen, Kopien usw.) die Benutzerberechtigung durch ein persönliches Paßwort im Rahmen einer Anmeldeprozedur (log in). Manche Programme verfügen über integrierte Sicherheitscodes mit bis zu zwölf Stellen. Als Paßwort beliebt, da leicht zu merken, sind Geburtsdaten, Vornamen oder gar Einzelzeichen (Sternchen) – allesamt leicht zu knacken, zum Beispiel mit „passwordcrakker"-Programmen.
Sinnvolle Vorschrift für die Gestaltung von Sicherheits-Codes: Sie sollten aus einer mindestens sechs- bis achtstelligen Kombination aus Buchstaben, Zahlen und Sonderzeichen bestehen, die in kürzeren Abständen geändert werden. Einen noch höheren Sicherheitsgrad erreichen Sie durch die Kombination der Benutzeridentifikation mit automatischen Benutzerprotokollen – auch hier gibt es Zusatz-Software.

Organisierte Datenträgerverwaltung

Manche Firmen inventarisieren und numerieren prinzipiell alle betrieblichen Datenträger vor der (quittierten) Ausgabe an die Benutzer und schreiben Datenträgerkataloge für jeden einzelnen PC vor. Sinnvolle Ergänzungen dieses Systems: Verbot der Benutzung anderer, vor allem privater Datenträger. Und die Vorschrift, Programme und Datenträger nach Dienstende nur unter Verschluß aufzubewahren.

Mechanische Sicherung

An erster Stelle steht die Sicherung von Räumen mit Computerausstattung durch Spezialtüren oder zumindest Sicherheitsschlösser. Eine Inbetriebnahme durch Unbefugte läßt sich durch Tastaturschlösser oder Schlüsselschalter an der Zentraleinheit verhindern.

Zu „Filterprogramme und Pipes unter MS-DOS"

Zu dem Satz „Diese Methode (Endeerkennung über Dateigröße) ist deshalb unbrauchbar." in MP 9/1988, Seite 266, muß folgendes bemerkt werden: Unbrauchbar ist natürlich nicht die Endeerkennung einer Datei mit Hilfe deren Größe, sondern die Programmierung eines Filters unter Verwendung der Geräte TRM:, CON:, KBD: bzw. der Standardannahme INPUT von TURBO-PASCAL, wegen des unbedingt erforderlichen Dateiendekennzeichens wie im Artikel ausführlich beschrieben.

MP

Programmpaket für Elektroinstallationshandwerk

Für PGHs und das Elektroinstallationshandwerk wurde ein Programmpaket in BASIC mit anschließender Compilierung für den ATARI 800XL/130XE geschrieben. Das Paket umfaßt:
- Rechnungsdeckblatt schreiben
- Berechnung der Regelleistung nach POA 564 und Vergleich mit Preisliste 44
- Pflegeprogramm für Preisdateien
- Materialberechnung und Abbuchung von der Materialdatei
- Pflegeprogramm für Materialdatei und Inventur.

An Hardware werden das Laufwerk 1050 und ein Drucker (1029/1027 oder Epson mit Interface) benötigt. 80 Prozent Arbeitszeiteinsparung werden damit erreicht.

Elektromeister Rainer Bock, Karl-Liebknecht-Str. 47a, Strehla, 8405; Tel.: 4 55 *Güldner*

Driver zur Kopplung DZT – AC 7100

Für die Ansteuerung des DZT 90×120/RS mittels AC 7100 wurde ein Programm in Turbo-Pascal geschrieben.
Es besteht aus dem eigentlichen Driver sowie mehreren Prozeduren zur komfortablen Bedienung des DZT aus Turbo-Pascalprogrammen. Sinngemäß ist die Lösung auf alle höheren Programmiersprachen übertragbar, welche einen Direktzugriff auf die E/A-Ports des Rechners gestatten (z. B. Basic unter SCP 1700).
Der Datenverkehr erfolgt über die V.24-Schnittstelle der ASP.

VEB Kombinat Schwermaschinenbau „Karl Liebknecht", Abt. WEV, Alt Salbke 6–10, Magdeburg, 3012, Tel. 4 32 28 20 *Rhode*

MC 80 mit 64 KByte RAM und Floppy Disk

Von uns wurde eine Lösung entwikkelt, die den Gebrauchswert eines MC 80 wesentlich steigert. Hierbei wurde der Hauptspeicher auf 64 KByte RAM erweitert. Das Betriebssystem sowie ein Floppy-Disk-Urladerprogramm befinden sich auf einer EPROM-Karte und werden automatisch nach Einschalten des Gerätes auf den RAM-Bereich kopiert. Nach jedem Reset wird das Betriebssystem mit Hilfe einer CRC-Prüfsumme auf Richtigkeit überprüft. Nur im Fehlerfall wird automatisch das Standardbetriebssystem vom EPROM nachgeladen.
Der Aufwand für die dazu notwendige Speicherbankverwaltung ist äußerst gering und wurde direkt am Koppelbus untergebracht. Speziell für den MC 80 existiert ein Floppy-Handler, der mit 4 Laufwerken arbeiten kann. Editor, BASIC bzw. beliebige Anwenderprogramme können sehr schnell von der Diskette in den RAM geladen werden. Im Zeichengenerator wurden zusätzlich die Kleinbuchstaben sowie die Umlaute implementiert. Ein leistungsfähiges Textprogramm ermöglicht die Textverarbeitung. Die Tastatur kann softwaremäßig in einen Schreibmaschinenmodus versetzt werden, indem mit der Floppy ein leicht geändertes Betriebssystem nachgeladen wird.
Zusammenfassend ergeben sich für die Neuentwicklung folgende Vorteile:
- hohe Flexibilität durch leichte Austauschbarkeit der Programme
- gute Ausnutzung des Hauptspeichers über 64 KByte (z. B. kann bei der Arbeit mit dem Editor der BASIC-Speicherbereich mitgenutzt werden)
- der Datenaustausch MC 80 – Bürocomputer bzw. MC 80 – Personalcomputer ist mit Hilfe der Floppy Disk möglich
- alle bisher erstellten MC 80-Programme sind ohne Änderung weiterverwendbar, da keine Softwareschnittstellen geändert wurden.

VEB Hartmetallwerk Immelborn, Am Bahnhof 5, Immelborn, 6207; Tel. Bad Salzungen 5 60 *Sandlaß/Graß*

Kalkulationsprogramm für KC 85/3

In Anlehnung an bekannte Tabellenkalkulationsprogramme (KP u. a.) wurde für den KC 85/3 ein Kalkulationsprogramm MINICALC entwikkelt. Das Programm ist in BASIC geschrieben und erbringt menügesteuert folgende Leistungen:
1. Definition einer Tabelle (max. 50 Zeilen, 20 Spalten, 3 Zeilen Überschrift) mit frei definierbaren Spaltenüberschriften und Zeilenbezeichnungen sowie Festlegung von Summierungen
2. Werteeingabe in die Tabellenfelder und Berechnen (falls gewünscht)
3. Ändern beliebiger Felder, Summierung, Löschen und Einfügen von Zeilen und Spalten
4. Betrachten frei wählbarer Spalten der Tabelle
5. Drucken frei wählbarer Spalten (z. B. 1, 3, 10, 4) in wählbarer Folge (Vorteil gegenüber den bekannten Tabellenkalkulationsprogrammen!) mit und ohne Summenzeilen
6. Speichern der Tabelle (1. Datei: Tabellenbeschreibung, 2. Datei: Tabellenwerte) auf MB-Kassette
7. Laden der Tabelle zur Weiterarbeit
8. HELP-Funktion.

Die Anwendung des Programms erfordert keine Programmierkenntnisse und ist durch die Menüführung noch einfacher als bei KP.

VEB Cottana Mühlhausen, ORZ, Johannisstr. 44, Mühlhausen, 5700 *Ulbrich*

Datenübertragung zwischen KC 85/2 (/3) und PC 1715

Es wurde ein Softwarepaket entwikkelt, das die Übertragung beliebiger Daten (also auch von Programmen, Texten, Bildern) zwischen PC 1715 und KC 85/2 (/3) ermöglicht.
Die Übertragung erfolgt seriell über einen V.24-Kanal, so daß außer einem V.24-Modul M003 für den KC 85/2 (/3) keine zusätzliche Hardware benötigt wird.
Die Datenübertragung kann auf Seiten des KC aus allen Programmiersprachen und Dienstprogrammen heraus mit Hilfe der vorhandenen SAVE- bzw. LOAD-Befehle gesteuert werden, beispielsweise mit CSAVE oder LIST#1 aus BASIC. Der dadurch eigentlich aufgerufene Zugriff zum Magnetbandgerät wird auf den V.24-Kanal umgelenkt.
Auf seiten des PC 1715 werden die Daten von in PASCAL geschriebenen Programmen in Empfang genommen und auf Diskette abgelegt oder umgekehrt.

Friedrich-Schiller-Universität Jena, Sektion Technologie für den WGB, Technikum LAURA, Ernst-Thälmann-Ring 32, Jena, 6900; Tel. 8 22 21 32 *Dr. W. Koch*

Druckprogramm PrintStar

Das unter CP/M, SCPX, CP/A, DCP, SCP 1700 u. a. laufende Programm für epson-kompatible Matrixdrucker erlaubt, nichtlateinische Schriften (Kyrillisch, Griechisch, Koptisch etc.) in WordStar- bzw. TP-Texten beliebig miteinander zu kombinieren. Mit einem menügesteuerten Full-screen-charakter-Editor lassen sich diese Zeichensätze modifizieren, aber auch eigene (z. B. Quasigrafikzeichen) generieren. Außerdem spricht das WS-kompatible Programm eine Vielzahl unterschiedlicher Druckmodi und Fähigkeiten des Druckers an, die WS nur zum Teil oder überhaupt nicht unterstützt: Mit CTRL-P-Befehlen (keine Escape-Folgen) kann man Elite, Pica-, Kursiv-, Entwurfs-, NLQ-, Eng-, Weit- und Proportionalschrift, Fett- und Doppeldruck, Hoch- und Tiefstellung, Unter- und Durchstreichung miteinander kombinieren. Zum Hervorheben selbstgenerierter Zeichen läßt sich die Unterstreichung verwenden. Daneben versteht das Programm auch alle übrigen CTRL-P-Befehle von WS (backspace, Überdrucken ganzer Zeilen, Druckstopp, Buchseitennumerierung, Seitenvorschub etc.). Andere CTRL-P-Befehle erlauben die Verwendung aller nationalen Zeichensätze des Druckers, einschließlich des IBM-Grafikzeichensatzes (wenn am Drucker vorhanden) und der selbstgenerierten Zeichen in einem Text. Allerdings erscheinen die Zeichen in ihrer generierten Form nur im Ausdruck; auf dem Bildschirm werden die entsprechenden ASCII-Zeichen dargestellt. Die Anzahl unterschiedlicher, in einer Textdatei verwendbarer Zeichensätze wird nur durch die Diskettenkapazität begrenzt. Neben der Verwendung aller WS-Punkt-Befehle für das Seitenlayout (Seitennumerierung, bedingter und unbedingter Seitenumbruch, Seitenlänge, header, footer, Anzahl der Kopf- und Fußzeilen, uni- und bidirektionaler Druck usw.), läßt sich der Zeilenabstand beliebig variieren und der linke Rand für Kopf- und Fußzeilen unabhängig von dem des Haupttextes einstellen. Es lassen sich mehrere Textdateien im Ausdruck aneinanderhängen. Nach dem Start des Programms vom Betriebssystem oder von WS aus kann man zwischen Einzelblatt und Endlospapier wählen. Die Seitenlänge wird automatisch auf das jeweilige Format eingestellt, wenn sie nicht durch einen Punkt-Befehl bestimmt wurde. Im Einzelblattmodus wird der Papierendefühler des Druckers abgeschaltet, um auch Formulare bis zum Seitenende beschreiben zu können. Der Ausdruck ist auf bestimmte Seiten beschränkbar, und die Anzahl der zu druckenden Kopien eines Textes läßt sich angeben. Nicht zuletzt minimiert ein je nach Betriebssystem bis zu 32 KByte großer Datenpuffer die Anzahl der Diskettenlesezugriffe und erlaubt ein entschieden schnelleres Drucken als mit WordStar.

Institut für Film, Bild und Ton, Abteilung SLE, Philipp-Müller-Straße, Block 17, Wismar, 2400 *Wolf*

dCOUNT – Häufigkeitsanalyse von dBASE-II-Dateien

Mit dCOUNT werden Häufigkeitstabellen von dBASE- oder REDABAS-Dateien (.DBF oder .DBD) in einem Durchlauf erstellt. Das Ergebnis kann mit Supercalc (oder KP) verändert und ergänzt werden.
Der COUNT-Befehl in dBASE II ermöglicht es, die Anzahl von Sätzen einer Datei entsprechend anzugebenden Bedingungen (FOR-Klausel) zu ermitteln. Dabei wird mit einem Durchlauf aber nur ein Zähler gesetzt. Für das Erstellen einer Häufigkeitstabelle sind deshalb ein großer Bedien- oder Programmieraufwand und eine lange Abarbeitungszeit erforderlich.
Für diesen Zweck wurde das Programm dCOUNT geschaffen. Durch eine Vorauswahl kann die Auswertung auf einen bestimmten Teil der Datei reduziert werden.
Das Auswerten der Datei, das Auszählen für die gesamte Matrix erfolgt in einem Durchlauf.
Die Ergebnismatrix kann man sich direkt als Textdatei ausgeben lassen und in eigenen Programmen weiterverarbeiten.
Normalerweise wird man sich eine Kommandodatei (.XQT) für Supercalc (oder wahlweise für KP) ausgeben lassen. Dadurch wird eine Supercalc-Tabelle mit den Ergebnissen, Beschriftungen sowie Summenzeilen und -spalten gefüllt. Vor dem Ausdruck können dann noch beliebige Veränderungen der Tabelle vorgenommen werden, z. B. andere Beschriftungen, Prozentwerte und andere Berechnungen.
Die Grenzen des Programms ergeben sich im wesentlichen aus dBASE und Supercalc. Das Programm wurde in Turbo-Pascal unter CP/A geschrieben.

Institut für Virale Zoonosen, Tornowstr. 21, Potsdam, 1561; Tel. 40 56 *Giard*



Börse

Kompatibilität zwischen C 128, PC 1715 und Schneider-PC

In vielen Einrichtungen werden zur Zeit verschiedene Rechnertypen eingesetzt. Einen großen Teil nehmen dabei die Typen PC 1715 (W), AC 7100, Schneider-PC 1512 sowie Commodore C 128 (D) ein. Für diese Rechnertypen wird die Kompatiblität durch das Diskettenformat **KAYPRO IV** erreicht. Dabei ist zu beachten, daß die zu benutzenden Disketten auf dem Commodore C 128 (D) mit dem Formatierungsprogramm **MFM-FORMATTER** initialisiert werden müssen. Danach ist ein problemloser Datenaustausch, z. B. zwischen TURBO-PASCAL und REDABAS, oder die Arbeit mit dem Textprozessor TP bzw. mit POWER möglich.
Das Format **KAYPRO IV** muß beim PC 1715 mit dem Programm FORMATP im Betriebssystem CP/A eingestellt werden, für den Schneider-PC 1512 ist **CONVERT** zu nutzen. Der mögliche Datenaustausch bezieht sich nur auf die Datenfiles, da unterschiedliche Betriebssysteme (CPM 3.0+, CP/A, CP/M) genutzt werden. Als COM-Datei (TURBO-Compiler, dBASE, WordStar) findet die für das jeweilige Betriebssystem konzipierte Software Anwendung.
Zur Nachnutzung werden angeboten:
- Beschreibung der durchzuführenden Konvertierungsschritte
- Formatierungsprogramm für C 128.

Technische Hochschule Leipzig, Sektion Elektroenergieanlagen, Wissenschaftsbereich Anlagentechnik, Karl-Liebknecht-Straße 132, Leipzig, 7030; Tel. 3 94 31 94 *Jannasch/Valtin*

Grafische Funktionsdarstellung mit A 7100

Der A 7100 verfügt über eine leistungsfähige s/w-Grafik, die mit der Grafikerweiterung SCP-GX nutzbar ist. Das vorliegende, mit TURBO-PASCAL erstellte Programm KKS.CMD ermöglicht die grafische Darstellung von Funktionskurven im I. und IV. Quadranten eines kartesischen Koordinatensystems. Nach dem Programmstart wird auf dem Bildschirm ein Koordinatenkreuz mit positiver x-Halbachse und y-Achse dargestellt. Die Teilung ist auf beiden Achsen mit 20 Pixel festgelegt. Da für die grafische Darstellung 400 × 400 Pixel verwendet werden, stehen auf der x-Achse 20 Teilstriche und auf der y-Achse ± 10 Teilstriche zur Verfügung. Auf der untersten Zeile des alphanumerischen Bildschirms werden das Menü und die Eingabeechos dargestellt. Aus dem Menü sind folgende Funktionen aktivierbar:
Werte laden (W): Ein mit seinem Namen zu spezifizierendes Datenfile (Endung DAT), das eine Wertetabelle enthält, wird gelesen und die entsprechende Funktionskurve dargestellt, indem die in der x-Koordinate aufeinanderfolgenden Punkte mit Geraden verbunden werden. Es können maximal 5 Funktionskurven, die durch ihre Linienart zu unterscheiden sind, übereinandergezeichnet werden.
Teilung (T): Der Maßstab der Achsen wird immer so festgelegt, daß die zur Verfügung stehende Fläche vollständig genutzt wird. Mit „T" können die Funktionskurven in x- und y-Richtung getrennt verkleinert werden.
Raster (R): Mit „R" wird ein Raster im Abstand der Teilstriche über das Koordinatensystem gelegt und damit das Ablesen von Funktionswerten erleichtert.
Druck (D): „D" ermöglicht eine Hardcopy des aktuellen Bildschirminhaltes auf einem grafikfähigen Drucker.
Save (S): Der Bildschirminhalt wird in ein File mit anzugebenden Namen (Name.PIC) gerettet. Das Dateiformat entspricht dem vom Grafikeditor GEDIT erzeugten Format.
LOAD (L): Eine Bilddatei (Name.PIC) wird gelesen und dargestellt. Es ist zwischen Überlagern und vorherigem Bildschirmlöschen zu wählen.

TU Dresden, Sektion Informationstechnik, Bereich Mikroelektroniktechnologie, Mommsenstr. 13, Dresden, 8027; Tel. 4 63 64 08 *Krenkel/Meyer*

Softwarelösungen zur Finanzplanung

Zur Anwendung liegen 3 Softwarelösungen für den PC 1715 vor. Sie umfassen das Erarbeiten und das Aufgliedern der
– Staatlichen Aufgaben (STAG)
– Staatlichen Auflagen auf Preisbasis 1 (STAL/PB1)
– Staatlichen Auflagen auf Preisbasis 2 (STAL/PB2)
sowie der Kennziffern des Finanzplanes durch
– das Ministerium auf die Kombinate
– die Kombinate auf die Betriebe
– die Betriebe auf Erzeugnisgruppen.
Die Softwarelösungen sind als Menüprogramme mit eigener Auswahl der Programmfolge aufgebaut. Sie wurden anwenderfreundlich angelegt, indem der Nutzer im Bildschirmdialog die verschiedenen Abarbeitungsvarianten selbst bestimmen kann.
Die vom Nutzer abgeforderten Eingabedaten werden in Dateien abgelegt und stehen für alle weiteren Rechenoperationen, Bildschirm- und Drucklistenausgaben mit aktuellem Stand zu Verfügung. Die Programme sehen mehrfache Korrekturmöglichkeiten bei der Eingabe und beim Aufgliedern der staatlichen Vorgaben in den verschiedenen Planungsphasen vor, das heißt, die auf dem Bildschirm angezeigten Tabellen lassen planungstechnische Eingriffe zu, bis das Aufgliedern vom Anwender selbst abgeschlossen wird.

VEB Forschung, Entwicklung und Rationalisierung des Schwermaschinen- und Anlagenbaus Magdeburg, Abteilung Ökonomische Grundsatzforschung, Bleckenburgstraße 25, Magdeburg, 3011; Tel. 4 42 81/84 *Pflug/Schulze*

Cross-System und Softwareemulator unter CP/M

Zur Erweiterung der funktionellen Möglichkeiten und der weitgehenden Gewährleistung der Aufwärtskompatibilität des AC 7100 zu CP/M-V2.2.-Rechnern wurden folgende Programmpakete entwickelt:
XSYS80 – ein Cross-System zur Generierung unter CP/M ladbarer Maschinenprogramme aus C-Quelltext oder 8080-Assembler. Es besteht aus folgenden Komponenten:
- **XC80** – ein Tiny-C-Compiler (erzeugt 8080-Mnemonics)
- **XASM80** – ein 8080-Cross-Assembler
- **XLOAD80** – ein Cross-Loader für CP/M.

Alle drei Komponenten sind selbst in C geschrieben und können folglich auf beliebigen Rechnern implementiert werden.
EMSYSZ80 – ein CP/M-Software-Emulationssystem für den Prozessor U 880; dient der Abarbeitung von Original CP/M-Programmen unter anderen Betriebssystemen und der Simulation einer entsprechenden Systemumgebung für diese Programme. EMSYSZ80 ist vorwiegend in 8086-Assembler geschrieben und ist an alle Rechner, die diesen Prozessor enthalten, ohne größere Probleme anpaßbar. Beide Programmpakete wurden unter CP/M 86 auf einem A 7100 mit Erfolg implementiert.

Technische Universität Karl-Marx-Stadt, Sektion Physik/Elektronische Bauelemente, PSF 964, Karl-Marx-Stadt, 9010 *Kaiser*

Arithmetikprogramm

Das Maschinenprogramm mit einer Länge von etwa 3 KByte realisiert die mathematischen Standardfunktionen und ist für die Verarbeitung von 5-Byte-Gleitkommazahlen (8 Dezimalstellen) in Maschinenprogrammen auf dem KC 85/2, /3 vorgesehen. Es läuft selbständig unter CAOS.
Zur Zeit wird an einer Einbindung in FORTH 880 gearbeitet. Das Programm steht für die Nachnutzung zur Verfügung.

SPS „Heinrich Hertz", Frankfurter Allee 14a, Berlin, 1035; Tel. 5 89 50 62 *Trappe*

V.24-Druckertreiber für A 7150 und K 8915

Der Anschluß eines Druckers an die V. 24-Schnittstelle (ASP) des A 7150 ist aufgrund der soft- und hardwaremäßigen Vorbereitung dieser Schnittstelle auf eine Nutzung für komplexe Datenübertragungssysteme nicht möglich.
Im Ing.-Büro für Rationalisierung wurde eine Lösung geschaffen, die bei minimalen Hardwareänderungen den Betrieb eines Epson LX 86 an dieser Schnittstelle ermöglicht. Das entwickelte Programm installiert nach Aufruf aus dem Betriebssystem einen Druckertreiber, der sämtliche Ausgaben an den Systemdrucker, an den die V.24-Schnittstelle angeschlossen ist, umleitet.
Außerdem steht ein Programm zur Verfügung, daß den Anschluß eines Druckers LX 86 an die KDS K 8915 ermöglicht, falls an dieser eine V.24-Schnittstelle vorhanden ist (ATS K 7028.20).

Ing.-Büro für Rationalisierung des VEB Kombinat Landtechnik Suhl im VEB KfL Sonneberg, Ackerstr. 40, PSF 2, Sonneberg, 6412; Tel. 4 51 81 App. 45 *Rockstroh*

Terminalemulation VT 100 unter MS-DOS

Terminalemulation ist eine Hard- und/oder Softwarelösung, welche das Terminalprotokoll eines Großrechners und die Schnittstelle zum Anwender terminalseitig auf einem selbständig arbeitsfähigen Rechner nachbildet und somit einem Nutzer die Abwicklung des Protokolls mit einer Großrechnerapplikation gestattet. Es wird ein Ressourcen- und Kommunikationsverbund zwischen Emulationsrechner und Großrechner erzielt.
Das Terminal VT 100 stellt einen Industriestandard der Firma DEC dar und ist für alle Großrechner mit DEC-ähnlicher Architektur einsetzbar.
Das vorliegende Emulationsprogramm realisiert die Emulation des genannten Terminals auf Rechnern der Klasse IBM-PC/AT mit einer Leiterkarte FPU als E/A-Koprozessorkarte. Das Softwareprodukt ist zu 90% in der Programmiersprache C implementiert und weist daher a priori eine hohe Kompatibilität auf. Durch klare Schichtung und Modularisierung ist die Software leicht änder- und erweiterbar. So kann ein Filetransferservice in kurzer Zeit installiert werden.
Für Rechner, die unter dem Betriebssystem MS-DOS arbeiten, aber nicht über eine Leiterkarte FPU verfügen, kann durch nachgeladene Driver das Verhalten derselben nachgebildet werden. Es liegt ein Driver für diese Aufgabe für die Rechner der Klasse IBM PC/XT vor. Aufgrund der Simulation der Hardware dieser Rechnerklasse beim EC 1834 ist derselbe Driver auch für diesen Rechnertyp einsetzbar. Auf den genannten Rechnerklassen wird die Emulation erfolgreich im Routinebetrieb genutzt. Ein Driver für den Rechnertyp A 7150 ist in Vorbereitung, wodurch das Softwareprodukt auf allen in der DDR derzeit verfügbaren MS-DOS-Rechnern einsetzbar wird.

TU Dresden, Informatikzentrum, Bereich SW, LFK BS, Mommsenstr. 13, Dresden, 8027; Tel. 4 57 55 61 *Oelsner*

Wir suchen ...

... eine Lösung zum Datenaustausch (Datenträger Diskette) zwischen C-64 und Schneider PCW 8258. Eine serielle Schnittstelle Schneider RS 232 C steht zur Verfügung. Verwendung finden Diskettenlaufwerke des Typs VC 1541. Erstrebenswert wäre eine Erweiterung des kompatiblen Datenaustauschs auf die Typen PC 1715 und AC 7100.

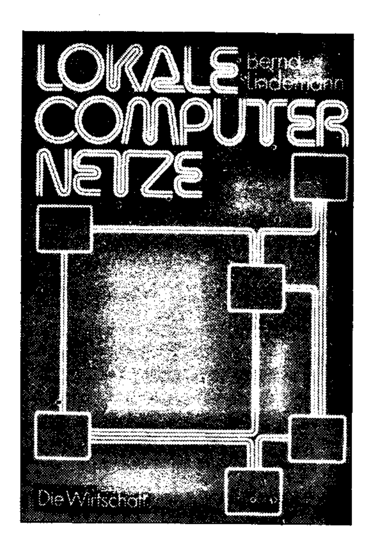
Staatlicher Forstwirtschaftsbetrieb Kyritz, Karnzow, 1901 *Kellas*

... eine Software/Hardware-Lösung zum Datentransfer zwischen dem Fernschreiber F 2000 und EC 1834 (Möglichkeit der Verarbeitung von FS-Texten mittels DCP-TP und umgekehrt).

VEB Chemieanlagenbaukombinat Leipzig-Grimma, Stammbetrieb Grimma, Dispatcherzentrale Kombinat, Grimma, 7240; Tel. 40 29 *Frenzel*

Lokale Computernetze

von B. Lindemann, Verlag Die Wirtschaft 1988, 244 Seiten, 49 Abb., 8 Tabellen, 14,– M



Die Broschüre gibt einen zusammenfassenden Überblick über den gegenwärtigen Entwicklungsstand auf dem noch sehr jungen Wissensgebiet der lokalen Netze (Local Area Network – LAN), wobei versucht wurde, alle relevanten Fragen der Anwendung über die Topologien, Übertragungsverfahren, Steuerverfahren usw. bis hin zur Struktur von Controllern und Fragen der Verwaltung von LAN zu betrachten. Es wird dabei auf einen breiten Leserkreis aus allen Bereichen der Volkswirtschaft orientiert, der sich privat oder beruflich einen Überblick über die Entwicklungen auf dem Gebiet moderner Kommunikationstechnologien verschaffen will. Diese Broschüre unterstützt somit aus inhaltlicher und preislicher Sicht bei der Schließung einer im Buchhandel vorhandenen Lücke.
Hauptanwendungsgebiete und Merkmale von LAN werden im ersten Abschnitt diskutiert. Im zweiten Abschnitt wird auf der Grundlage internationaler Dokumente eine Definition für LAN gegeben, es werden ihre Komponenten betrachtet und ihre Abgrenzungen zu anderen Kommunikationstechnologien diskutiert. In den folgenden drei Abschnitten werden Vermittlungstechnologien, Netztopologien, Übertragungsverfahren und Steuerverfahren für LAN betrachtet. Im sechsten Abschnitt wird auf der Grundlage des ISO-Referenzmodells eine Einführung in die Schichtenarchitektur von LAN und im siebenten Abschnitt eine Einführung in die Hardwarekomponenten für LAN gegeben. Der achte Abschnitt ist Fragen der Verkopplung von LAN mit gleichartigen bzw. anderen Netzen gewidmet. Im neunten Abschnitt werden konkrete z. T. herstellerspezifische LANs, wie Ethernet, Rolanet 1, IBM-Token-Ring, Wang Net und Hyperchannel, überblicksmäßig betrachtet. Der abschließende zehnte Abschnitt enthält eine wertvolle Zusammenfassung des gegenwärtigen Standes der Standardisierung von Netzen.
Zusammenfassend kann positiv eingeschätzt werden, daß der Autor versucht hat, eine sehr breite Palette an internationalen Entwicklungen auszuwerten und in die vorliegende Broschüre einzubeziehen. Ein wesentlicher Nachteil ist jedoch, daß fast alle „Blüten", die die LAN-Entwicklung in den letzten 15 Jahren hervorgebracht hat, kritiklos und gleichberechtigt nebeneinander gestellt werden. Die Folge ist, daß der Leser mit einer Vielzahl von Begriffen konfrontiert wird, die bei dem Umfang der Broschüre dann nicht hinreichend oder gar nicht definiert werden. Man gerät dadurch beim Lesen sehr schnell in Widersprüche, da die „Begriffswelten" der verschiedenen Hersteller bzw. Entwickler von LAN nicht widerspruchsfrei sind bzw. im Widerspruch zu früheren Aussagen in der Broschüre stehen. Der Hauptmangel ist folglich, daß der Leser nicht klar erkennen kann, was die Hauptentwicklungslinien auf dem Gebiet der LAN sind, die sich seit Mitte der Achtziger Jahre herauskristallisiert haben. Zweifellos wäre ein kritischeres Herangehen des Autors an die Veröffentlichungen der Jahre 1978 bis 1983 in diesem Sinne wünschenswert gewesen.

Prof. Dr. Thomas Horn

Computer Graphics
Gerätetechnik, Programmierung und Anwendung graphischer Systeme

von J. Encarnacao und W. Straßer. Akademie-Verlag, Berlin 1988

Mit Übernahme der 2. Auflage dieses Buches vom Oldenbourg Verlag ist dem Akademie-Verlag Berlin ein guter Griff gelungen. Auf etwa 450 Seiten behandelt es die Grundlagen der grafischen Datenverarbeitung in erschöpfendem Maße. Den Hauptteil bilden Gerätetechnik, grafische Systemprogrammierung mit GKS und 3D-Geometrieverarbeitung. Eingehende Behandlung finden hier die Beschreibung dreidimensionaler Objekte, ihre Transformationen sowie ihre Darstellung auf ebenen Ausgabemedien bis hin zu Visibilitätsverfahren und Beleuchtungsmodellen. Es geht also um solche Informatik-Grundlagen, die heute u. a. dafür sorgen, daß man synthetische Computerbilder kaum noch von Fotografien unterscheiden kann. Wichtige Randgebiete wie Dialog, Bildverarbeitung und CAD/CAM werden in den Zusammenhang von Computer Graphics eingeordnet.
Das Buch wendet sich an Studenten einschlägiger Fachrichtungen, an Hersteller von grafischen Hard- und Softwaresystemen sowie an Anwender, die den Prinzipien der grafischen Datenverarbeitung auf den Grund gehen möchten. Dieser Zielgruppe gemäß ist der Stil sachlich und trotz strenger Wissenschaftlichkeit gut verständlich. Anschaulichkeit wird erreicht durch zahlreiche Abbildungen, Fotos sowie einen originellen Farbbildteil. Ein sorgfältig ausgebauter und offenbar im Hochschulbetrieb wohlerprobter Anhang aus Übungen, Lösungen und einem Glossar unterstützt den Lehrbuchcharakter. Es mag sein, daß mancher Computeranwender die fertigen Quelltexte zum Abtippen vermißt, wie er sie in anderen Büchern zur gleichen Thematik findet. Dafür ist es nun mal ein Grundlagenwerk, und es wird dieser Rolle in Inhalt, Herangehensweise und Ausstattung in solider Weise gerecht. Nicht ganz zu teilen ist die Auffassung der Autoren im Vorwort, daß nach stürmischer Entwicklung des Fachgebietes in 10 Jahren seit der 1. Auflage nun eine zeitlos stabile Version erreicht sei. Sicher wird in den folgenden 10 Jahren die Entwicklung mindestens so stürmisch weitergehen – sowohl bei den Grundlagen (z. B. Animation, Wissensverarbeitung, neue Standards) als auch bei den Anwendungen (z. B. Robotersimulation, Verschmelzung von Grafik- und Textverarbeitung, Werkstückmodellierung). Gerade deshalb darf man zufrieden sein, auf einem derart bewegten Gebiet den gegenwärtigen Stand des Grundlagenwissens so umfassend vorzufinden wie in diesem Buch.

Dr. Günter Rothhardt

CAD-Grundlagen

von R. Helmrich und P. Schwindt, VEB Fachbuchverlag Leipzig 1986, Lizenzausgabe des Vogel-Buchverlag Würzburg, 212 S., 30,– M

Das leicht lesbare und gut verwertbare Buch von Helmrich und Schwindt hat einen weit gefaßten und zu anspruchsvollen Titel. Erst der Untertitel „Eine Anleitung zum rechnergestützten Zeichnen" engt gebührend ein. Aber auch der Ausdruck „Grundlagen" ist zu global. Tatsächlich werden nur *Grundbegriffe* des rechnergestützten Zeichnens allein mechanischer Gebilde behandelt.
Das Buch ist eine Brücke für jene, die sich in dieses Gebiet *einführen* lassen wollen. Außer den unmittelbar fachlichen Informationen zur rechentechnischen Hard-/Software-Basis von CAD-Systemen und den elementaren zeichen- bzw. rechentechnischen Beschreibungen geometrischer Gebilde beschäftigt es sich auch mit Anforderungen an die Qualifizierung von CAD-Systemnutzern.
Bemerkenswert ist, daß zahlreiche Aufgaben und nachfolgend entsprechende Lösungen zu allen relevanten Kapiteln vergeben werden. Damit wird das Buch zum autodidaktischen Lehrbuch.
Die kategorische Aussage jedoch (S. 15): „Maschinen können nicht denken, dieser Grundsatz wird bestehen bleiben" – ist unzulässig und grob irreführend. Unbestreitbar entwickelt sich insbesondere mit der 5. Rechnergeneration ein leistungsfähiges „maschinelles Denken", von dem ganz besonders für das qualitative Niveau künftiger CAD-Systeme bedeutende Auswirkungen erwartet werden. So ist auch der Satz (S. 157): „Die Übertragung von Konstruktionswissen auf das CAD-System ist auch in Zukunft kaum zu erwarten" nicht mehr zulässig.
Sehr viele der im Buch behandelten Begriffe haben bleibenden Wert, wie Koordinatensystem, geometrische Operationen, Menü, Konstruktion, Zeichnungselemente, Modell, Ebene, Raum, Körper u. a. Sie haben bereits ein gutes Alter. Aber gerade die durch den Rechner bedingten Leistungen und Begriffe unterliegen einem bemerkenswerten Wandel. Der rechnergestützte Entwurf (CAD) steht ganz am Anfang. So ist auch diese CAD-Einführung nicht ein Standardwerk, sondern eine zeitgemäße, nützliche Darstellung – die bereits viele Leser gefunden hat. Es ist zunehmend notwendig, sich mit diesen technischen Mitteln engagiert zu beschäftigen. Im vorliegenden Beispiel wäre das Zeichnen elektrischer/elektronischer oder informatischer Gebilde gut gewesen. Aber dies gehörte nicht zur Absicht der Autoren.

Prof. Dr. Dr. M. Roth

Transistor- und Schaltkreistechnik

von H.-J. Fischer und W. E. Schlegel, 3., völlig überarbeitete Auflage, Militärverlag der DDR, Berlin 1987, 386 Seiten, 456 Bilder, 14,30 M

Die stürmische Entwicklung der Nachrichtentechnik und Elektronik erfordert, neben dem beruflich Interessierten auch zunehmend dem Amateur Möglichkeiten bereitzustellen, sich sowohl schnell als auch umfassend über diese Gebiete zu informieren.
Diesem Charakter entspricht das vorliegende Buch an sich, und dieser Band entspricht dem um so mehr, da er nun schon in dritter, völlig überarbeiteter, auf den neuesten Stand gebrachter Auflage vorliegt.
Dem Amateur ist es ein moderner Leitfaden für die Praxis. Das Buch verzichtet aber auch nicht auf einen kurzgefaßten theoretischen Teil. Obwohl aus der Amateurbibliothek herausgegeben, so ist es doch kein fertiges „Kochbuch" für den Amateur. Vielmehr wird dem Amateur ein Wissen durch die Autoren aufbereitet, das er für den Aufbau funktionstüchtiger Schaltungen benötigt. Dazu werden neben der Vorstellung charakteristischer Schaltungsbeispiele auch theoretische Grundlagen und technologische Zusammenhänge vermittelt. Damit wird dem Amateur das Rüstzeug zum Verständnis weiterführender Spezial- und Fachliteratur geboten.
So wird der Mikroprozessor und dessen Applikation aus diesem Buch bereits ausgeklammert und auf die dazu existierende, sehr umfangreiche Spezialliteratur verwiesen.
Mit seiner Praxisorientierung ist dieser Band dem Amateur, aber auch dem Ingenieur ein wertvolles Standardwerk.

Dr. I. Schreiber

Expertensysteme
Lösung nichtformalisierter Aufgaben im Dialog mit dem Computer

von E. W. Popow, Verlag Nauka, Moskau 1987, 288 Seiten, in russischer Sprache

Das Buch richtet sich an Informatiker, an Spezialisten auf den Gebieten der Künstlichen Intelligenz und der automatisierten Steuerungssysteme. Die

systematische Darlegung macht es aber auch empfehlenswert für Studenten der entsprechenden Fachrichtungen und für Interessenten, die bereits eigene Erfahrungen zur Informatik besitzen.
Als Teilgebiet der Künstlichen Intelligenz (KI) stellen sich die Expertensysteme (knowlegde engineering) die Aufgabe, die Prinzipien und Instrumentarien der KI in die Lösung schwieriger Anwendungsprobleme einzubringen, die ein Expertenwissen verlangen.
Das Buch ist eine der wenigen Quellen, in der die seit der Herausbildung des Wissensgebietes Expertensysteme erzielten Ergebnisse und Erfahrungen systematisch und bewertend dargelegt werden.
In der Einführung werden die Grundlagen, die Architektur und die Methodologie zur Erarbeitung von Expertensystemen aufgeführt. Dabei wird eine große Anzahl von Expertensystemen und Instrumentarien vorgestellt. Das System MYCIN zur Behandlung von Infektionskrankheiten dient in den weiteren Kapiteln neben anderen als Bezug zur Darlegung der Funktionsweisen und Konstruktionsprinzipien der Expertensysteme.
Den Problemen Wissensrepräsentation, Dialogführung mit dem Nutzer, Suchstrategie für Lösungen, Steuerung der Funktionsweise und Erklärkomponenten ist je ein Kapitel gewidmet. Der Wissenserwerb ist in zwei Kapiteln beschrieben.
Das Literaturverzeichnis mit 184 Quellenangaben im Zeitraum von 1976 bis 1986 gibt einen repräsentativen Überblick.
Hervorzuheben ist das Bemühen des Autors, durch systematische Gliederung, Erläuterung der verwendeten Begriffe und zahlreiche Beispiele einen umfassenden Einblick in die Problematik zu geben. Aufgeführte Lösungsvarianten zu jedem Teilproblem werden vom Autor begründet und bewertet, wobei auch auf perspektivische, aber noch nicht ausreichend erforschte Lösungswege hingewiesen wird. Besonderes Augenmerk wird auf Reihenfolge und Inhalt der einzelnen Entwicklungsstufen bei der Erarbeitung eines Expertensystems gelegt. Obwohl sich die Beispiele hauptsächlich auf Expertensysteme auf der Grundlage von Produktionssystemen (Wissensbasis sind Regeln und Fakten) stützen, sind die Anforderungen an die Funktionsweise der einzelnen Komponenten allgemein formuliert. Die Kapitel zum Wissenserwerb werden am Beispiel des Regel- und Faktenerwerbs im System TEIRESIAS abgehandelt und bieten Ansatzpunkte für weiterführende Forschungen auf der Basis von Frame-Strukturen und Vererbungshierarchieen der Wissensbasis.
Einschränkend muß auf das Fehlen eines Sachregisters und auf die Vielzahl der verwendeten Abkürzungen hingewiesen werden. In den Kapiteln zur Dialogführung und zu den Erklärkomponenten werden zur grafischen Kommunikation keine Darlegungen gemacht. Hier beschränkt sich der Autor leider nur auf die natürlichsprachige Verständigung (russisch), deren Probleme selbst nur wieder im Rahmen eines Expertensystems zu lösen wären.

Dr. K. Biedka

Das Software-Lexikon

von Klaus W. Jamin, Expert Verlag Ehningen und Taylorix Fachverlag Stuttgart 1988, 350 Seiten

Die rasante Entwicklung der Computertechnik ist verbunden mit einer Flut ständig neuer Begriffe, Produktnamen, Schlagwörter. Sowohl Anfänger wie Spezialisten dürften daher dankbar sein für ein Lexikon, das ihnen hilft, bei dieser Informationsfülle die Übersicht zu behalten. Die Autoren des vorliegenden Buches wollen sich vor allem an den Praktiker wenden und den, der es werden will. Großer Wert wurde daher auf eine einfache und anschauliche Erläuterung der Begriffe gelegt. Allerdings wird bei vielen Beispielen dieses Buches die Problematik einer solchen Verfahrensweise deutlich: Die Exaktheit leidet doch hin und wieder darunter. Beispielsweise bei der Bestimmung des Begriffs AI (artificial intelligence), der lediglich anhand der Meinungsäußerungen prominenter US-amerikanischer Wissenschaftler vorgenommen wird. Hier ist übrigens auch das Fehlen eines Querverweises zur Künstlichen Intelligenz zu bemängeln, wie insgesamt Verweise recht lax gehandhabt werden.
Bei der Auswahl der über 1000 Stichwörter lag der Schwerpunkt auf den Programmiersprachen, wobei auch Beispiele aus der praktischen Anwendung von Programmen berücksichtigt wurden. Auf die Erläuterung von Begriffen wie Luftcomputer oder Krankencomputer hätte dabei getrost verzichtet werden können.
Der unbestreitbare Vorteil des Buches liegt vor allem in seiner Aktualität. Wenn man berücksichtigt, daß viele der aufgenommenen Begriffe erst im letzten Jahr entstanden sind oder geprägt wurden – beispielsweise gilt 1987 als **das** Jahr des Desktop Publishing (DTP), und im April 1987 stellte IBM ihr Personal System /2 mit dem Betriebssystem OS/2 vor – dann setzt die Herausgabe eines solchen Lexikons zu Beginn dieses Jahres Maßstäbe.

Hans Weiß

Telekommunikation – Netze und Dienste der Deutschen Bundespost

von A. Albensöder (Hrsg.) R. v. Dekker's Verlag, Heidelberg 1987, 179 S., 63 Bilder, Literatur- und Stichwortverzeichnis

Die Telekommunikation wird im zunehmenden Maße Bestandteil der Infrastruktur. Voraussetzung der Anwendung sind Fernmeldenetze und -dienste.
Die vorliegende Veröffentlichung beschäftigt sich in gut verständlicher Form mit bestehenden Fernmeldenetzen wie Telefonnetz, integriertes Text- und Datennetz (IDN) sowie Breitbandverteilnetze. Schwerpunkte bilden dabei Datexdienste mit Leitungsvermittlung und Paketvermittlung. Ein besonderes Kapitel ist den Endgeräten der Fernmeldenetze einschließlich der der Breitbandverteilnetze und Glasfaser-Overlaynetze gewidmet. Von besonderem Interesse erscheinen jedoch die Ausführungen zu Weiterentwicklungen der Fernmeldenetze und Fernmeldedienste. Sie beziehen sich vor allem auf den Übergang von der analogen zur digitalen Telefonübertragungs- und -vermittlungstechnik sowie auf die Breitbandverteilnetze. Wichtig für bestimmte Anwendergruppen könnten die Ausführungen zum dienstintegrierten digitalen Fernmeldenetz ISDN sein, wofür ein digitales Telefonnetz Voraussetzung ist. Informationen zu Planungen bestimmter Etappen der Entwicklung mit Jahresangaben ihrer Realisierung gestatten einen Einblick in die Entwicklung von Netzen und Diensten:
Digitales Telefonnetz (ab 1985/86, Übertragungs- und Vermittlungstechnik); ISDN (ab 1988, 64 kbit/s-Dienste); Integration schmal- und breitbandiger Fernmeldedienste der Individualkommunikation im Breitband-ISDN (90er Jahre); Integration der Individualkommunikations- und Verteildienste im Glasfaser-Universalfernmeldenetz (langfristige Option).

Prof. Dr. W. Schoppan

Desktop Publishing: Setzen und Drucken in eigener Regie

von Gerhard Bader, Vogel-Buchverlag Würzburg, 1987, 186 S., ISBN 3-8023-0195-1

Das Buch gibt einen guten Überblick über eine neue Technologie der Herstellung von Drucksachen. Die immer besseren grafischen Möglichkeiten moderner 16- und 32-Bit-PC sowie die rasante Entwicklung der Druckertechnologien haben aus der einfachen elektronischen Textverarbeitung ein ganz neues Gebiet der Computeranwendung mit einem zunehmend großen Marktanteil entstehen lassen. Bilder und Texte können zu einem Dokument verbunden, frei gestaltet und in hoher Qualität vom einfachen Matrixdrucker bis zu professionellen Laserbelichtungsanlagen ausgegeben werden. Mögliche Anwendungen sind dabei die dezentrale Drucksachenherstellung in Betrieben bis zu kleinen und mittleren Druckereien.
Das Buch, das fast vollständig mit dieser Technologie (DTP) hergestellt wurde, gibt Informationen über Hard- und Software, auch über ihre Vor- und Nachteile. Es wird eine auch für den unerfahrenen Leser verständliche Einführung gegeben. Von besonderem Interesse für Einsatzvorbereitung und Entscheidungsfindung ist das Eingehen auf die Schnittstellenproblematik zu Standardsoftware, Textsystemen und CAD-Programmen. Der Autor beschreibt ausführlich die wichtigsten Softwareprodukte und ihre praktische Anwendung, macht Möglichkeiten und auch derzeitige Grenzen deutlich. Ein Anhang mit Abbildungen ausgedruckter Beispiele und einem Verzeichnis der Fachbegriffe bildet eine sinnvolle Ergänzung der Textaussagen.
Die Einbindung von DTP in Robotron-Systeme und DDR-Standardsoftware wie TP, REDABAS, MULTICAD usw. ist denkbar und teilweise bereits erprobt (siehe MP 4/88).

Matthias Fischer

Maschinensprache des IBM-PC in der Praxis

von Isa Brors, Hüthig Verlag, Heidelberg 1986, 278 Seiten, 20 Beispiele

Der vorliegende Band befaßt sich mit der Assembler-Programmierung des IBM-PC/XT und wendet sich damit an Nutzer, denen die Möglichkeiten einer höheren Programmiersprache als nicht ausreichend erscheinen bzw. an solche, für die Assembler-Programmierung einziges Mittel zur Lösung ihrer speziellen Probleme ist.
Nach einer knappen Einführung über die Darstellung von Zahlen in verschiedenen Zahlenbereichen wird kurz die weitere Struktur des 8088-Prozessors behandelt. Dabei werden insbesondere die möglichen Adressierungsarten erläutert, wobei die durchgängige Verwendung einer exakten und einheitlichen Terminologie wünschenswert wäre. Im dritten Kapitel wird der komplette Befehlssatz des 8088 in knapper und übersichtlicher Form beschrieben. Die hier gewählte Darstellungsform eignet sich gut für die Nutzung des Bandes als Nachschlagewerk. Die sich anschließenden beiden Kapitel über die Arbeitsweise von Assembler, Linker und Debugger – es wird auf MASM, LINK und DEBUG Bezug genommen – erläutern den prinzipiellen Aufbau eines Assembler-Programmes sowie die notwendigen Hilfsmittel und Arbeitsschritte zum Erzeugen eines lauffähigen Programms. Ein erstes Programmbeipiel illustriert diese Erörterungen. Das zehnte Kapitel beschreibt anhand von zwanzig Programmbeispielen mit den dazugehörigen Listings sowohl die Arbeitsweise der funktionellen Einheiten eines PC (Bildschirm, Tastatur, Drukker usw.) als auch häufig angewandte mathematische Verfahren und Sortieralgorithmen. Einige dieser Beispiele sind durchaus geeignet, Bestandteil einer nutzereigenen Modulbibliothek zu werden. Insgesamt sind die gewählten Beispiele konstruktiv und sorgfältig programmiert. Die letzten beiden Kapitel vermitteln überblicksmäßig Informationen über Dateiformate und Hardware-Erweiterungen.
Die im Anhang aufgelisteten Angaben zu den wichtigsten Interrupts und Code-Tabellen sowie ein Register und ein Schnell-Index zum Auffinden der Assembler-Befehle erleichtern die Nutzung dieses Bandes als Kompendium.
Die Autorin hat sich auf die wesentlichen Aspekte der Assembler-Programmierung beschränkt, wobei bei einigen Beispielen weitergehende Erläuterungen zum mittlerweile Standard gewordenen PC-Betriebssystem PC-DOS/MS-DOS wünschenswert wären.
Ich bin der Auffassung, daß der mit den Grundbegriffen der Mikrorechner-Programmierung vertraute Leser sich mit Hilfe der kurzen und übersichtlichen Ausführungen dieses Bandes einen Überblick über die Möglichkeiten der Assembler-Programmierung des 8088 verschaffen kann.

Bernd Höppner

EISA contra Mikrokanal

Einen neuen Busstandard für 32-Bit-Personalcomputer zu schaffen haben sich führende Computerproduzenten vorgenommen. Auf einer gemeinsamen Pressekonferenz am 13. September in New York gaben die Firmen AST, Compaq, Epson, Hewlett Pakkard, Olivetti, NEC, Tandy, Wyse und Zenith Einzelheiten zu dem Vorhaben bekannt. Ausgangspunkt der Entwicklung ist zum einen die Unzufriedenheit der Hersteller mit der technischen Qualität des Mikrokanals, den IBM mit dem PS/2 als neuen Industriestandard durchsetzen möchte. Andererseits sicher das Verhalten des Marktführers, das den anderen Firmen den Zugang zum Mikrokanal über Lizenz- und Kennummernvergabe erschwert. Demzufolge haben sich inzwischen bereits mehr als 50 weitere Hersteller zur Unterstützung der Extended Industry Standard Architecture (EISA) bereit erklärt, unter ihnen beispielsweise Intel, AT&T, Chips & Technologies, Microsoft, Borland und Ashton-Tate. Die ersten Lizenz-Mikrokanal-PC-Produzenten Dell, Tandy und Apricot wollen, wie die meisten anderen auch, allerdings zweigleisig fahren und beide Architekturen bei künftigen Entwicklungen berücksichtigen.
Kernpunkt des neuen Busses ist seine Abwärtskompatibilität mit dem bisherigen AT-Bus (auch ISA – Industry Standard Architecture genannt) im Gegensatz zum Mikrokanal. Während IBM meinte, die Möglichkeiten des AT-Busses seien ausgeschöpft und eine neue Architektur notwendig, demonstrieren die kompatiblen Busse Smart Slot Architektur von AST, Flex-Architektur von Compaq und der Nubus beachtliche Leistungsfähigkeit. Wie die Tafel zeigt, hat auch EISA gegenüber der Mikrokanal-Architektur (MCA) nicht nur den Vorteil der Kompatibilität; in fast allen Parametern kann dem Mikrokanal Paroli geboten werden. Dazu kommt die von den EISA-Initiatoren immer wieder hervorgehobene Offenheit des Systems.
Von Intel sollen bis zum März 1989 die ersten EISA-kompatiblen Steuerchips kommen, Chips & Technologies sowie Western Digital planen ebenfalls kompatible Chipsets, so daß etwa Ende 1989 mit der Vorstellung der ersten EISA-PCs zu rechnen sein wird. *MP – We*

Funktion und Leistung im Vergleich

Merkmal	*EISA*	*MCA*
– 32-Bit-Adressierung und Daten	ja	ja
– Bus-Master-Unterstützung	ja	ja
• maximale Burst-Rate	33 MByte/s	20 MByte/s
• automatischer Datenpfad	unterstützt	nicht unterstützt
• Bus-Master-Chip	verfügbar	nicht verfügbar
– DMA-Unterstützung		
• 32-Bit-Adressierung	unterstützt	nicht unterstützt
• 32-Bit-Datenverarbeitung	unterstützt	nicht unterstützt
• Burst-Übertragung	unterstützt	unterstützt
• max. DMA-Übertragungsrate	33 MByte/s	5 MByte/s
– Interrupt-Auslösung	Pegel- oder Flankentriggerung	Pegeltriggerung
– max. unterstützte RAM-Größe	>16 MByte	16 MByte
– softwaremäßiges Konfigurieren von Erweiterungsplatinen	ja	ja
• Adapter-Kennung	vom Hersteller bestimmt	von IBM bestimmt
• Unterstützung vorhandener Platinen	ja	nein
– Platinengröße	406 cm²	232 cm²
– maximale Anzahl von Steckplätzen	15	8

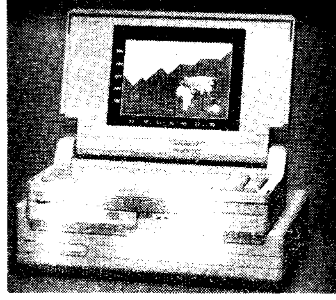
Nachfolger von Framework II

Als Nachfolger des integrierten Softwarepaketes Framework II kommt von Ashton-Tate jetzt Framework III auf den Markt. Es wurde in den Funktionsbereichen Textverarbeitung, Tabellenkalkulation, Datenbank, Grafik und Datenfernübertragung wesentlich verbessert und erweitert; Dateien des Vorgängers können jedoch problemlos übernommen werden.
Die prinzipielle Bedienung des Programmes hat sich nicht geändert. Alternativ zur Tastatur gibt es jetzt auch eine Maus. Die Textverarbeitung wurde mit einer automatischen Silbentrennung versehen. Ebenfalls automatisch werden nunmehr Inhaltsverzeichnisse erstellt. Die Verwaltung von Fußnoten, die bis zu 64000 Zeilen lang sein dürfen, ist jetzt möglich. Vordefinierte Abkürzungen werden auf Tastendruck automatisch ausgeschrieben. Insgesamt stehen zwölf Wörterbücher für die Rechtschreibung zur Auswahl.
Framework III kann nun neben dem gewohnten Delimited-Format Datenbankframes zur weiteren Bearbeitung mit dBase im dBase-eigenen DBF-Format speichern.
Die Rechengeschwindigkeit der Tabellenkalkulation konnte um den Faktor zwei gesteigert werden. Die Neuberechnung einer Tabelle kann man nun auf Teilbereiche beschränken.
Damit sich Framework III nahtlos in bestehende Systeme integrieren läßt, liest und schreibt es Dateien in folgenden Formaten: Mulimate II, Lotus 1-2-3, Wordperfect, Multiplan, Wordstar und ASCII. Auch die Übernahme von Framework-Daten in die Grafik-Software Chart-Master wird jetzt angeboten.
Für den Einsatz als Einplatzlösung setzt Framework III einen IBM-PC/XT/AT oder PS/2 oder 100prozentig kompatible Systeme mit 640 KByte Hauptspeicher voraus. Dabei sind zwei Diskettenlaufwerke im 3,5-Zoll- beziehungsweise 5,25-Zoll-Format oder ein Diskettenlaufwerk mit Festplatte notwendig. Framework III verlangt als Betriebssystem PC-/MS-DOS 2.11 oder höher. Für den Einsatz in Netzwerken gibt es die LAN-Version, die fünf Lizenzen enthält. *MP*

IBM-PC wieder mit AT-Bus

Offenbar unter dem Druck des Marktes und gegen die bisherige Strategie seit der Vorstellung des PS/2 stellte IBM im September einen neuen kompatiblen Personalcomputer mit AT-Bus vor, der aber bereits Merkmale des PS/2 aufweist. Das **Modell 30286** hat eine 80286-CPU, VGA-Grafik und 3,5-Zoll-Disketten. Der RAM umfaßt bis zu 4 MByte auf der Platine. Mit diesem Modell soll dem bisherigen AT-Nutzer der Übergang auf das PS/2 vermutlich erleichtert werden. *MP*

SLT/286: VGA-Grafikmodus nun auch auf LC-Bildschirm



Dem Trend zu den sogenannten Schoßcomputern, den Laptops, hat jetzt auch die Firma Compaq Rechnung getragen. Während sie bereits als führend bei den Portables galt – Erstanbieter eines tragbaren PCs mit 20-MHz-80386-Prozessor – bietet sie mit dem SLT/286 nun erstmals die Möglichkeit, auf einem LC-Display in Supertwisted-Technik den von IBM für das PS/2 entwickelten leistungsfähigen VGA-Grafikmodus darstellen zu können. Die Auflösung beträgt dabei 640 × 480 Pixel, und die Farben werden in 8 Graustufen umgesetzt. Für die Darstellung der Farben kann natürlich ein externer Farbmonitor angeschlossen werden.
Weitere Merkmale des Computers sind die Modularität – das Bild zeigt den SLT/286 beispielsweise mit der von unten angesetzten Erweiterungseinheit, die Tastatur ist abnehmbar –, ein wechselbarer Akkublock, 80C286-Prozessor mit 12 MHz, 640 KByte RAM, bis 3,64 MByte ausbaufähig, 20- oder 40-MByte-Festplattenlaufwerk, 3,5-Zoll-Diskettenlaufwerk mit 1,44 MByte. Die 34,2 × 10,5 × 21,5 cm³ große Systemeinheit (B × H × T) wiegt 6,4 kg. *MP*

Neue Turbo-Versionen von Borland

Von dem bekannten Softwareproduzenten kommen jetzt die neuen Compiler Turbo-Pascal 5.0 und Turbo-C 2.0. Die MS-DOS-Übersetzer verfügen erstmals über einen integrierten Quellcode-Debugger und werden nun von Turbo-Assembler und Turbo-Debugger unterstützt.
Bei **Turbo-Pascal 5.0** halten die aus Version 4.0 verbannten Overlays wieder Einzug. Im Gegensatz zu den 3.0-Overlay-Dateien werden diese Programmabschnitte nun als Pascal-Unit deklariert und können – ebenso wie der Programmeditor – wahlweise im EMS-Speicher arbeiten.
Turbo-Pascal 5.0 speichert zusätzliche Symboltabellen für den externen Turbo-Debugger. Das Hilfsprogramm verwendet dabei ein anderes Tabellenformat als MS-Codeview. Programme, die mit Microsoft-Compilern übersetzt wurden, lassen sich jedoch über ein Konvertierungsprogramm auf das Borland-Symbolformat ändern und damit untersuchen. Um Debugger-Abstürze zu vermeiden, arbeitet das Programm im Remote-Betrieb von einem zweiten Rechner aus oder schützt sich durch den Protected-286-Modus vor agressiven Bugs.
Die Verbesserungen von **Turbo-C 2.0** gegenüber Release 1.5 betreffen vor allem den integrierten Debugger. Ebenso wie bei Pascal 5.0 lassen sich Unterbrechungspunkte setzen, die Belegung von einzelnen Variablen und Strukturen beobachten und ändern. Der Compiler soll etwa ein Viertel schneller übersetzen als sein Vorgängermodell 1.5. *MP*

UNIX-Standard in Japan mit dem Sigma-Projekt

Ähnlich der X-Open-Gruppe und der Open Software Foundation (OSF) bemühen sich rund 40 japanische Firmen mit Unterstützung der Regierung um eine Standardisierung des Betriebssystems UNIX. Das Sigma (Software Industrialized Generator and Maintenance Aids system) genannte Projekt soll eine Kompatibilität zwischen den Produkten unterschiedlicher Hersteller auf Quellcodeniveau schaffen. Als Teil dieses Projekts werden auch die Workstations geschaffen, die mit dem Sigma Operating System (Sigma-Betriebssystem) arbeiten. Das Sigma-Operating System basiert auf dem UNIX-System V Release 2 (mit Berkeley-Erweiterung 4.2 BSD). Es beinhaltet die UNIX-Erweiterungen für Kommunikation und Grafik und wird mit einer Japanisch-Erweiterung ausgestattet. Hardwareseitig verlangt die Workstation-Spezifikation von Sigma eine 32-Bit-CPU und einen IEEE-Gleitkommaprozessor auf der CPU-platine.
Das Sigma-Projekt wird unter anderem von so renommierten Firmen wie Fujitsu, Hitachi, Sharp, NEC und Toshiba unterstützt. *MP*

Kontostände per Fingertipp

Im IBM-Labor in Böblingen wurde ein neuer berührungsempfindlicher Bildschirm, der IBM 4737 entwickelt, der sich besonders für den Einsatz in Schalterhallen und Foyers von Banken und Sparkassen, bei Versicherungen, im Einzelhandel und im Dienstleistungsgewerbe eignen soll. Mit dieser neuen Informations- und Serviceeinheit kann der Kunde neben den bekannten Kontostandsabfragen und Barabhebungen auch Geldüberweisungen und Daueraufträge einrichten, Börsen- oder Finanzierungsangebote abfragen, sich über Immobilienangebote informieren, Reisen und Hotelzimmer buchen oder Mietwagen reservieren. Durch Antippen bestimmter Felder oder Bilder auf dem Bildschirm wird der Computer gesteuert.
Die verwendete Technologie basiert auf druckempfindlichen Sensoren aus piezoelektrischem Material. Dazu wurde vor der Bildschirmröhre eine Glasplatte angebracht, die die gleiche sphärische Geometrie wie die Röhre hat. Zwischen Röhre und Platte befindet sich an jeder Ecke jeweils ein druckempfindlicher Sensor aus piezokeramischem Material. Wird die Glasplatte nun mit leichtem Druck berührt, werden die Kräfte mit Hilfe der Piezoelemente gemessen. Aus dem Verhältnis läßt sich die Lage des Berührungspunktes errechnen.
Der IBM 4737 wird von einem PC gesteuert. Auf ihm sind die Grafikprogramme und die Dateien gespeichert. Die Ein- und Ausgabefunktionen und die Datenverschlüsselung sind auf einem speziellen Zusatzrechner gespeichert. Vertrauliche Informationen werden mit dem DES (Data Encryption Standard)-Algorithmus verschlüsselt.

Quellen: BZB. Sachmagazin. – Hamburg 91 (1988) 6. – S. 25
IBM-Nachrichten. – Stuttgart 38 (1988) 293. – S. 70 Wi

Leistungssteigerung durch variable Architektur

Der „Verwaltungsaufwand" (sog. Overheads), der für die Realisierung einer Aufgabe auf einer EDVA notwendig ist, hat maßgeblichen Anteil an der Leistung einer Anlage. Bei Spezialrechnern, die für ein bestimmtes Problem konzipiert sind, ergibt sich ein geringer Verwaltungsaufwand. Die Bereitstellung von Spezialrechnern für jedes Problem ist aus Fertigungsgesichtspunkten nicht effektiv.
Eine Lösung für dieses Problem bietet die Firma UMA-Elektronik an. Das Konzept beruht auf einem Baukastenprinzip, mit dem entsprechend dem Problem Rechner aufgebaut werden, ohne jedoch Eingriffe und Veränderungenin der Hardware vorzunehmen. Die Bausteine sind auf Platinen in konventioneller Technik mechanisch miteinander verbunden. Mittels einer speziellen Transsoftware werden ohne äußeren Eingriff im Bedarfsfall die optimalen Schaltungen erzeugt. Dadurch soll der Durchsatz bis zu einem Faktor 40 erhöht worden sein. Die Aufgabenstellungen werden in kleine Teilaufgaben zerlegt, die durch logische Verschiebung der Hardware-Software-Schnittstelle immer mit einer optimalen Hardware in Form variabler Parallelrechner gelöst werden. Somit wird auf der Hardware-Ebene die Funktion ausgeführt, die ein Compiler vergleichsweise auf der Ebene einer Programmiersprache ausführt. Es entfallen die laufenden Wiederholungen von Fetch-Operationen, die in allen Computern der von-Neumann-Architektur auftreten.
Voraussetzung der variablen Architektur sind keine neuen Bauelemente, sondern lediglich abänderbare Gatter-Anordnungen (AGAs) und dazugehörige Entwicklungswerkzeuge (AGA-CAD). Die abänderbaren Gatter können mit hoher Geschwindigkeit umprogrammiert werden (5000 mal pro Sekunde). Dem Hauptprozessor eines bestehenden Computersystems werden ein oder mehrere AGAs zugeordnet. Die Verarbeitung der Datenmengen erfolgt mit einfachen Algorithmen. Große Datenblöcke werden bewegt, indem die Adressierung des Speichers geändert wird. Die Daten bleiben in den RAM-Zellen, liegen jedoch für den Prozessor in einem anderen Adressenbereich. Die Umadressierung erfolgt unabhängig von der Datenmenge in einer konstanten Zeit. Dadurch werden bei großen Datenmengen erhebliche Zeiten gespart.
Die Abänderung der Hardware erfolgt durch Änderung der Daten in den RAM-Zellen der AGAs, die eine Änderung der Logikstruktur bewirken.

Quelle: eee. Elektronik-Technologie ... – Leinfelden-Echterdingen (1988) 12. – S. 23–24 Wi

Hochauflösende Farb-LCD

Das neue Verfahren der japanischen Firma Dai Nippon Printing Co. soll dazu beitragen, die Auflösung von Farbfilterschichten, die in Flüssigkristall-Displays verwendet werden, zu verbessern. Die Bildelemente (Grundfläche $10 \times 10 \mu m^2$) sollen mit der neuen Technik mit einer Genauigkeit von $\pm 2 \mu m$ auf dem üblichen Glassubstrat erzeugt werden.
Für die neuen Filter werden Pigment-Dispersionen auf der Basis von Plastikmaterial verwendet. Die Pixel-Dreiergruppen müssen aus den Farben Rot, Grün und Blau bestehen.
Mit diesem neuen Verfahren soll sich die Zahl der Bearbeitungsschritte um 80% verringern.
Die ersten Muster dieser hochauflösenden Farb-LCDs mit 5 und 6 Zoll Diagonale wurden bereits ausgeliefert. Bis Jahresende will man mit der Serienproduktion beginnen und anschließend Typen mit 14 Zoll Diagonale fertigen.

Quelle: Elektronik. – München 37 (1988) 12. – S. 7 Fa

Laserdrucker ohne Laser

Eine Neuheit auf dem Gebiet der Laserdrucker stellt ein von der amerikanischen Firma Westinghouse Electric entwickeltes System dar, bei dem statt der bisher üblichen Laser ein Elektrolumineszenz-Emitter benutzt wird. Damit entfallen die sonst erforderlichen Umlenkspiegel und alle anderen mechanisch bewegten Teile.
Der Elektrolumineszenz-Emitter wird mit den üblichen Verfahren der Dünnschichttechnik hergestellt. Im Gegensatz zu anderen Elektrolumineszenz-Lichtquellen tritt an einer Kante des Elements Licht aus.
Diese neuartigen Emitter können zu linearen Arrays zusammengefaßt werden, die dann über die gesamte Breite der Druckfläche reichen.

Quelle: Elektronik. – München 37 (1988) 12. – S. 7 Fa

PenWriter – elektronische Schreibtafel von Scriptel

Eine Neuentwicklung auf dem Gebiet elektronischer Digitalisierungseinrichtungen (Geräte zur Eingabe grafischer Informationen in Computer) stellt die elektronische Schreibtafel PenWriter der Firma Scriptel Corporation (Ohio) dar. Bei dieser Schreibtafel handelt es sich um ein Computerzusatzgerät, mit dem man handgeschriebene Informationen mit Hilfe eines Computers schreiben, verschicken oder speichern kann.
Der PenWriter ist ein 12-Zoll-Schreibbildschirm mit einem dazugehörigen Stift, der an einen IBM- oder IBM-kompatiblen Computer angeschlossen werden kann. Dabei können drei verschiedene Arten von Bildschirmen verwendet werden:
– die von hinten beleuchtete Fluoreszenz-LCD
– die von hinten beleuchtete Elektrolumineszenz-LCD
– die Neongas-„Plasma"-Anzeige mit einem Auflösungsvermögen von 640 × 400 Pixel.
Mit dem Abtaststift, der mit einer Metallspitze versehen ist, kann man Notizen und Informationen aller Art direkt auf den Schirm schreiben. Damit entfällt das Eintasten der Informationen. Wörter und Zahlen können in maschinengeschriebene Daten umgewandelt, gespeichert oder auf einen anderen PenWriter übertragen werden.
Für den PenWriter ist jede PC-Software unter MS-DOS geeignet. Anwendungsgebiete sind u. a. das Ausfüllen von Formularen, das Erkennen von Unterschriften oder die computergestützte Konstruktion.

Quelle: Newsweek vom 30. 5. 1988 Wi

Erste optische ICs

An der Entwicklung rein optisch arbeitender integrierter Schaltungen arbeitet das amerikanische Joint-Venture-Unternehmen Photonic Integration Research Inc., an der die japanischen Firmen NTT und Mitsubishi sowie das Battelle Memorial Institute beteiligt sind.
Bei diesen optisch arbeitenden Schaltungen wird auf ein Silizium-Substrat eine Schicht aus einem Silikat-Material gelegt. Dieses Material entspricht im wesentlichen dem, das sonst für die Herstellung von Glasfasern verwendet wird. Mit herkömmlichen Photolithografie- und Ätzverfahren werden anschließend die (optischen) Wellenleiterbahnen erzeugt.
Die ersten Bauelemente sollen optische Schalt- und Verzweigungselemente, Multiplexer und Demultiplexer sowie Schaltermatrizen (Kreuzpunktschalter) für die Kommunikationstechnik sein.
Die optischen ICs sollen vor allem die Aufgabe übernehmen, die bei der elektrischen Schaltungstechnik die Leiterplatte hat. Die eigentlichen aktiven Elemente (Lichtemitter bei Wellenlänge über 1100 nm und Detektoren) lassen sich aus Silizium nicht herstellen. Sie müssen auf dem Siliziumchip extra aufgesetzt werden. Diese Technik ist aber noch in der Entwicklung. Wenn künftig GaAs-Substrate verwendet werden, kann eine monolithische Integration erfolgen.

Quellen: Elektronik. – München 37 (1988) 13. – S. 7
Elektronik. – München 37 (1988) 14. – S. 7 Fa

Optische 4×4-Matrixschalter

Schalter in gegenwärtig übliche Lichtleitersysteme einzufügen, stellte bisher ein Problem dar. Die schwedische Firma L. M. Ericsson entwickelte den Prototyp eines optisch arbeitenden Schalters auf der Basis von Lithiumniobat. Dieser Matrix-(Kreuzpunkt)-Schalter kann maximal 4×4 Signalwege schalten, und die bei diesen Elementen kritische Übersprechdämpfung konnte auf 50 dB erhöht werden. Außerdem müssen die zu schaltenden optischen Signale nicht polarisiert sein. Die bisherigen Erwärmungs- und Geometrie-Einschränkungen sollen weitgehend verringert worden sein.

Quelle: Elektronik. – München 37 (1988) 13. – S. 7 Fa

Festplattenkarte über 100 MByte

Die Firma SCSE bietet jetzt eine Leiterkarte mit einem integrierten Festplattenlaufwerk an, das die 100-MByte-Grenze überschreitet.
Die Wincard 100 belegt 1½ Steckplätze und bietet 104 MByte formatierte Speicherkapazität, 28 ms mittlere Zugriffszeit, PC/XT/AT- und 386-Interface mit PC-Bus/SCSI-Host-Adapter und einen Interleave-Faktor von 1,1 bei einer Leistungsaufnahme von 11 Watt. Die Wincard 100 kann als zweite oder dritte Festplatte auch bei einem bereits installierten Festplattencontroller zur Systemerweiterung eingesetzt werden. MP

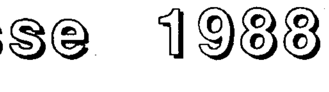
Taschencomputer befolgt Sprachkommandos

Der von der Texlon Corp. entwickelte batteriegespeiste Taschencomputer PTC-755 soll auf gesprochene Befehle reagieren und bei der Durchführung von Inventuren eingesetzt werden. Der Taschencomputer ist mit einem Zweiwege-Spracherkennungsbauelement ausgerüstet. Der Benutzer liest die Menge und den Strichcode einer Ware in das Mikrofon. Die Information wird vom Computer wiederholt und auf einem 16-Zeilen-Bildschirm angezeigt, um eine nochmalige Korrektur zu ermöglichen. Danach wird die Information abgespeichert.
Die Firma Texlon testet gegenwärtig das Gerät bei Inventuren in Lagerhäusern, beim Wareneingang und bei der Warenauffindung.

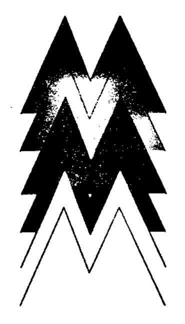
Quelle: Design News vom Mai 1988 Wi



Bericht



Leipziger Herbstmesse 1988



Unter dem traditionellen Motto „Für weltoffenen Handel und technischen Fortschritt" waren in Leipzig vom 4. bis 10. September wiederum etwa 6000 Aussteller aller Kontinente vertreten. Der Schwerpunkt dieser Messe lag zwar bei der integrierten Meß-, Prüf- und Regelungstechnik – kein Schwerpunkt für die MP – dennoch können wir im folgenden einige Exponate vorstellen, die für unsere Leser sicher von Interesse sind.

Beispielsweise zum Thema Desktop Publishing (DTP) – von traditionellen Fotosatzanlagen-Herstellern oft lieber als Electronic Publishing bezeichnet, während Vertreter der Seybold Computer Publishing Conference im September wiederum den umfassenderen Begriff Computer Publishing bevorzugten.

Die Firma Linotype war mit ihrer kompletten **Lichtsatz-Serie 2000** vertreten: Die Serie 2000 (Farbbild 1; alle Farbbilder siehe 2. Umschlagseite) ist eine neue Workstation-Familie für alle Satzanwendungen von Text- und Bilderfassung über Satzgestaltung und Text-/Bildintegration bis hin zur Belichtung der Druckfolie bzw. Ausgabe auf Laserdrucker. Herzstück ist hierbei eine AT-kompatible Workstation (Tower rechts im Bild). Die Workstation 2100T für die Texterfassung verfügt über einen Arbeitsspeicher von 512 KByte RAM, die Workstation 2200C für Satzgestaltung und 2200G für Text-/Bildintegration sind mit jeweils 2 MByte RAM ausgestattet. Standardmäßig sind alle drei Modelle mit einem 1,2-MByte-Diskettenlaufwerk (5¼") ausgerüstet. Die Workstation 2200C verfügt zusätzlich über eine 40-MByte-Festplatte, die Workstation 2200G über eine 70-MByte-Festplatte. Beide beinhalten auch einen 90-MByte-Streamer (Magnetband).

Das nicht ausgestellte Nachfolgemodell, Serie 3000, kann sogar drei Festplattenlaufwerke mit je 300 MByte aufnehmen.

Für die Serie 2000 stehen zwei 14"-Monitore zur Verfügung, ein Bildschirm mit 720 × 350 Bildpunkten und 76 Graustufen (Bildfrequenz 70 Hz) und für die Workstation 2200G optional ein Greyscale-Monitor mit 720 × 350 Bildpunkten und 256 Graustufen (Serie 3000: 19"-Bildschirm mit 2048 × 1024 Punkten). Die Bildintegration kann mit dem Scanner 20 (Bildmitte) für Strich- und Halbtonoriginale bis zum Format 300 × 425 in einer Auflösung von 300 bis 900 dpi (Punkt pro Zoll) oder durch Übernahme von Bilddaten (Grafiken oder Illustrationen) von einem Apple-Macintosh oder einem IBM-kompatiblen PC erfolgen.

Für die Erzeugung des PostScript-Formates zur Ausgabe der Seiten auf Laserbelichter Linotronic 300 (links im Bild) oder Apple-LaserWriter II (Bildmitte) steht der Raster-Image-Processor RIP 2 zur Verfügung (linker Tower).

Während der LaserWriter II mit einer Auflösung von 400 dpi für Korrekturausdrucke oder Desktop Publishing (Publizieren auf dem Tisch) Verwendung findet, werden mit den Laserbelichtern Linotronic 100, 500 und 300 Fotopapier, Filme oder Direktdruckfolien mit einer Auflösung von 312 bis 2500 dpi belichtet. Beispielsweise wurde das Messesignet am Anfang unseres Berichtes mit einem Macintosh erstellt und über den RIP 2 auf dem LaserWriter II ausgegeben.

Die Firma Compugraphic, vor über 20 Jahren in Boston gegründet und seit kurzem eine 100prozentige Tochter von Agfa, gilt als einer der bekanntesten Hersteller von Fotosatzsystemen. Der gegenwärtigen DTP-Welle kommt Compugraphic mit dem „Einstiegsmodell" PTS Publisher entgegen, einem Wyse-PC 286, CRT-Fotosatzbelichter MCS 8000, Laserdrucker CG 300 und dem notwendigen Softwarepaket. Als anspruchsvolleres Werkzeug wurde der **MCS Integrator Serie 19** gezeigt (Farbbild 2), das Mitglied einer Familie von Workstations, die unter dem Betriebssystem UNIX Mulitasking- und Mehrplatzbetrieb erlauben. Basierend auf der PowerView-Produktlinie kann auch mit Maus und Menü-Technik gearbeitet werden. WYSIWYG-Darstellung, interaktive Seitenlayout-Gestaltung und Bearbeitung von Linien, Layout-Gitter und Netzwerk-Optionen sind weitere Leistungsmerkmale. Hardwarebasis sind die Systemeinheit mit Prozessor 80286, 4,6 MByte RAM, 20 MByte Festplatte (optional 40 MByte), 1,2 MByte (optional 360 KByte) Diskette sowie Scanner, Belichter oder Laserdrucker.

Als Tochterfirma von HELL stellte elaplan buchholz zwei Systeme für Texterfassung bzw. Desktop Publishing aus. **INPUT** heißt ein System zur Texterfassung und -bearbeitung durch PCs am Satzsystem DOSY. Die am Input-PC erfaßten Texte können direkt an das DOSY überstellt werden (V.24) oder direkt auf einen Drucker oder auf Diskette ausgegeben werden. Im Farbbild 3 sind neben dem DOSY-Sichtgerät 2069 (links) als Input-PCs ein EC 1834 von Robotron (Mitte) und ein PCD-2 von Siemens zu sehen. Im Vergleich zum DS 2069 ist beim Input-PC keine Darstellung von vier Schriften möglich. Alle Standardzeichen und spezielle Sonderzeichen werden als Ersatzzeichen dargestellt. Es wird keine Spezialtastatur, sondern die Standard-PC-Tastatur verwendet. Alle Sonderzeichen, Befehle und Befehlsketten können frei programmierbar auf die Funktionstasten gelegt werden.

Weiterhin bot elaplan buchholz das Satzsystem **TOPSET** (Farbbild 4) an. Das System TOPSET besteht aus dem Erfassungs- und Bearbeitungsplatz auf VENTURA-Basis (links im Bild), dem Laserdrucker (im Bild oben) für die Ausgabe von Korrekturabzügen und die Wiederholung mehrerer Seiten als Kleinauflage sowie dem Laserbelichter DESKSET. Während der Laserdrucker A4-Seiten mit einer Auflösung von 300 dpi in etwa 10 Sekunden beschreibt, benötigt der Laserbelichter für eine A4-Seite etwa 30 Sekunden (bei 600 dpi). Weiterhin beinhaltet das System TOPSET den Datenconverter und Übertragungsrechner DESKCOPY zur Ausgabe der Produktion als Strichbild und die Faxstrecke MINIFAX.

Im Rahmen der Interscola stellte Robotron den **Bildungscomputer A 5105** als Steuerrechner für Schülerexperimente aus (Farbbild 5). Diesen Rechner haben wir bereits näher in MP 10/1988 vorgestellt. Ebenfalls in der Interscola war ein kompletter **CAD-Arbeitsplatz** von Robotron-Elektronik-Dresden zu finden (Bild 1). Er besteht aus einem A 7100, A 7150 (im Bild) oder EC 1834, einem grafischen Tablett K 6405.00/.20, einem Plotter K 6418/11 sowie einem Drukker K 6313/14. Weiterhin wurde ein nicht ausgestellter Laserdrucker (RANK XEROX 4045 Plus) angeboten. Als Software ist der Allgemeine Grafische Editor GEDIT/M16 mit zwei Ebenen, der Benutzeroberfläche und dem grafischen Editor, im Angebot. Als Betriebssysteme können DCP 1700, Ausgabe 3.2, bzw. DCP 3.2 Verwendung finden.

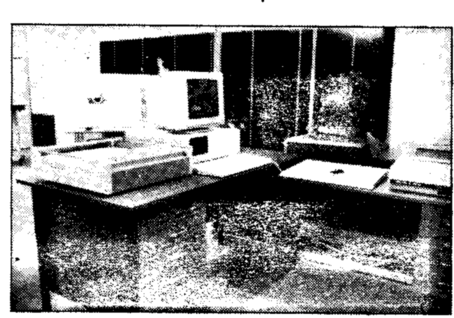
Im sowjetischen Pavillon wurden ein Drucker und ein Plotter der Saratower Filiale der Kammer für Handel und Industrie der UdSSR ausgestellt. Der elektrostatische Drucker **ESPU-K** hat eine Druckbreite von 256 mm, eine Auflösung von 5 Punkten pro mm und eine Geschwindigkeit von 150 Zeilen pro Sekunde (Bild 2).

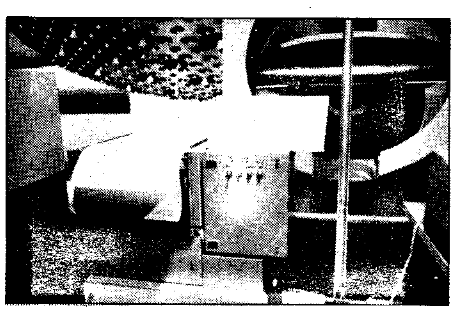
Der Farbplotter **CGP-1** kann Seiten bis zu einem Format von 1600 × 256 mm ausgeben. Mit einer Auflösung von 5 Punkten pro mm wird ein einfarbiges Bild in 5 Minuten erzeugt. Vier Farben können nacheinander ausgegeben werden (Bild 3).

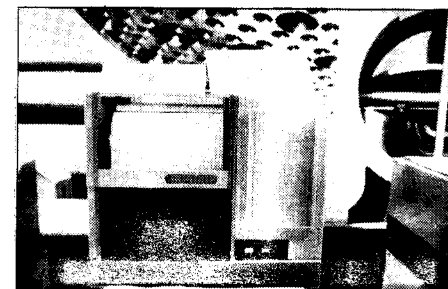
Als nützliches Hilfsmittel, um Monitordarstellungen in Echtzeit einem größeren Publikum vorführen zu können, gibt es seit kurzem von verschiedenen Herstellern für Tageslicht-Projektoren LC-Displays. Diese werden auf die Projektionsfläche aufgelegt, an den PC angeschlossen und wirken, vom Licht des Projektors durchleuchtet, wie eine veränderbare Folie. Die von Kodak vorgestellte Projektionsplatte **DATASHOW** (Farbbild 6) hat die Abmessungen 300 × 330 × 53 mm^3 (das Display etwa 200 × 150 mm^2) und erreicht eine Auflösung von 640 × 200 Pixel. Das Bildseitenverhältnis ist 1,3 : 1. Die Verbindung zum IBM PC, XT, AT oder einem Kompatiblen wird über die serielle Schnittstelle hergestellt. Weitere Merkmale: Split-screen-Modus, in dem zwei verschiedene Bildschirminformationen miteinander verglichen werden können; Bildumkehr, mit der positive oder negative Darstellung möglich ist; elektronische Zeiger; Infrarot-Fernbedienung.

Ein weiteres LC-Display, **Geha data vision**, wurde von der Handelsgesellschaft Transcommerz angeboten (Farbbild 7). Auch dieser Flüssigkristall-Bildschirm hat eine Auflösung von 640 × 200 Pixel. Die Abmessungen sind 364 × 343 × 52 mm^3 (Bildgröße 208 × 130 mm^2 mit 25 Zeilen zu 80 Zeichen). Das Bildseitenverhältnis beträgt 2 : 1. Die Kopplung ist mit jedem IBM- oder kompatiblen PC über RGB-Anschluß möglich. Das Display erlaubt ebenfalls die Wahl zwischen Positiv- und Negativdarstellung, außerdem gibt es eine Freeze-Funktion zum Einfrieren des Bildes.

Transcommerz zeigte weiterhin einen Videoprinter, mit dem Videosignale binnen 17 Sekunden in scharfe, kontrastreiche Papierbilder, auch mit







Halbtönen, umgesetzt werden können. Der **Mitsubishi P70B** (Farbbild 8) arbeitet nach dem Thermoverfahren und kann die Bilder, etwa A4-Format, vorwärts oder um 180 Grad gedreht ausdrucken. Ein Bild kann beliebig oft ausgegeben werden. Die Auflösung beträgt 640 Punkte × 576 Zeilen in einer 64stufigen Grauskala; die Darstellung ist positiv und negativ möglich. Der Anschluß erfolgt über RGB/TTL, FBAS, BNC oder Centronics-Parallel-Interface; es lassen sich die Zeilennormen nach PAL/SECAM und NTSC nutzen. Es ist also die Kopplung von Monitoren, PCs, professionellen Video-Recordern, Fernsehgeräten und Videokameras möglich. Damit ist der Videoprinter geeignet zum Einsatz in allen Bereichen, in denen medizinische Geräte, Meß- oder Prüfgeräte und Computer arbeiten.
Sharp ist in Leipzig seit Jahren mit Heimelektronik vertreten, unter anderem mit programmierbaren Taschenrechnern. Neben diesen wurde nun das leistungsfähigste Modell der Pocketcomputer, der **PC-1600 (Farbbild 9)**, vorgestellt. Der PC-1600 hat 16 KByte RAM (etwa 11 KByte frei verfügbar), 96 KByte ROM, BASIC-Programmierung, ein Display mit 4 Zeilen zu 26 Zeichen (156 × 32-Punktmatrix) sowie RC-232-, SIO- und Analogschnittstelle. Das Besondere am PC-1600 ist, daß er sich mittels verschiedener Module zu einem System zusammenstecken läßt. Dies sind beispielsweise ein Vierfarb-Plotter CE-1600 P für A4-Format und ein 2,5-Zoll-Floppylaufwerk mit 64 KByte pro Seite. Darüber hinaus lassen sich Disketteneinheiten, RAM- und ROM-Module sowie Drucker auch extern betreiben.
Mit der bisher üblichen, verwirrenden Mehrfachbelegung von Tastaturen soll die von Hohe Electronics entwickelte Multifunktions-Tastatur **The Board** Schluß machen (Farbbild 10). Bei ihr sind auf einer Reihe von Tasten anstelle der üblichen Beschriftung LC-Displays aufgebracht, die das jeweils aktuelle Zeichen anzeigen. Die Matrix von 20 × 8 bzw. 12 × 8 Punkten erlaubt die Darstellung der verschiedensten Zeichen und Symbole, die per Software binnen Millisekunden verändert werden können. Damit entfallen die lästigen Schablonen und Tastenetiketten. Hardwarebasis sind ein integrierter 8-Bit-Mikroprozessor, 64 KByte RAM sowie 16 bis 32 KByte EPROM. Interessant ist die String-Taste, mittels der ganzen Zeichenketten mit bis zu 126 Anschlägen auf jeder einzelnen Taste abgelegt werden können, beispielsweise Befehlsketten, Textbausteine oder Steuerungssequenzen. The Board gibt es in verschiedenen Ausführungen, die sich alle im wesentlichen an das Layout der IBM-Standard-Tastatur MF1 halten, darüber hinaus jedoch zusätzliche programmierbare Funktionstasten und Softwareunterstützung bieten. In Leipzig wurde das Ergebnis einer Zusammenarbeit der Firma Hohe Electronics mit der Karl-Marx-Universität Leipzig gezeigt, das Programmsystem CAP/DS (Computer Aided Planning/Decision Support System). Es erlaubt auf der Basis der linearen Optimierung mittels einer Komplexmethode eine computergestützte Planung von optimalen Veränderungen im Produktionssortiment und die Planung der Technologiestruktur von Industriebetrieben. Der Einstieg in dieses System wird bei Verwendung der Tastatur so einfach und benutzerfreundlich, daß auch ohne Computervorbildung nach kurzer Einarbeitungszeit mit dem Programmsystem gearbeitet werden kann.

Text und Fotos: Hans Weiß/Herbert Hemke

2. Internationale ATARI-Messe



Als vollen Erfolg wertet Alwin Stumpf, Geschäftsführer der ATARI Computer GmbH und damit gleichzeitig Veranstalter, die 2. Internationale ATARI-Messe, die vom 2. bis 4. September in Düsseldorf stattfand. Mit 111 Ausstellern aus dem In- und Ausland, das sind 41 % mehr als im Vorjahr, und einem neuen Besucherrekord (26000 Besucher gegenüber 20000 im vergangenen Jahr) belegte diese Messe das kontinuierlich zunehmende Interesse an allen ATARI-Produkten.
Im Mittelpunkt standen überwiegend professionelle Branchenlösungen aus allen Anwendungsbereichen, industrielle Lösungen und der Midi-Sektor, der ebenfalls sehr stark vertreten war. Besonderen Anklang bei allen Beteiligten und Besuchern fanden auch in diesem Jahr wieder die zahlreichen Workshops mit ihrem breiten Themenspektrum und die Präsentation im zentralen Forum.
Auf reges Publikumsinteresse stieß vor allem das neue ATARI-Desktop Publishing System. Auf der Basis des ATARI Mega ST und der DTP-Software Calamus bietet dieses System eine professionelle Lösung; zusätzlich war eine Druckmaschine installiert, auf der die mit dem ATARI DTP-System erstellten Seiten sofort gedruckt wurden.
Zahlreiche Neuentwicklungen unterstrichen, daß ATARI bestrebt ist, auch im professionellen Markt in immer mehr Bereiche einzudringen. Das zeigten während der Messe neu vorgestellte Netzwerklösungen ebenso wie ein 44-MB-Wechselplattensystem, ein zum Mega ST kompatibler portabler Rechner, der Einsatz von 19"- und 24"-Großbildschirmen oder das Polaroid Palette System für die ATARI ST-Computer. Insgesamt reichte die Spannweite bei den Ausstellern von Computerspielen bis hin zur firmenspezifischen CIM-Strategie und von der Textverarbeitung bis hin zur kompletten mehrplatzfähigen Branchenlösung.
„Nach den Messeerfolgen in diesem und dem vergangen Jahr," erklärte Alwin Stumpf abschließend, „steht für uns fest, daß die ATARI-Messe zu einer bleibenden Einrichtung wird. Für unsere Marketingstrategie ist der direkte Kontakt zum Anwender eine der tragenden Säulen."

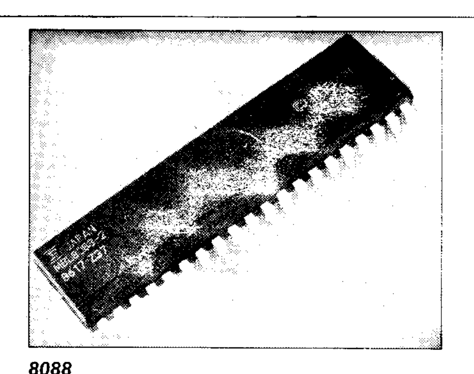
Fach- und Informationstagung FORTH

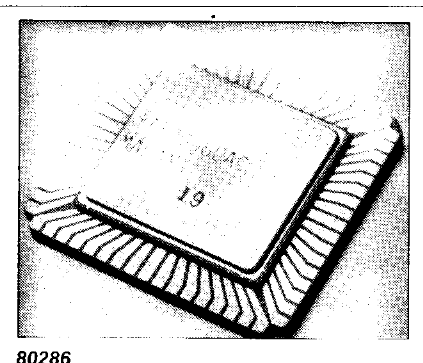
Die Kammer der Technik Suhl, BfS Elektrotechnik/Elektronik, IG FORTH, hatte zur Informations- und Fachtagung FORTH 1988 vom 20. bis 22. April 1988 nach Suhl eingeladen. Unter sehr guten organisatorischen Bedingungen (Dank dafür gilt dem KDT-Bezirksverband Suhl) wurden etwa 190 Teilnehmer in 23 Vorträgen über Systementwicklungen, Anwendungen und Erfahrungen zu FORTH informiert. Im Mittelpunkt des Plenarvortrages (*Vack*/ZKI – Internationaler Stand zu FORTH – wie geht es weiter?) standen die Aspekte
- Durchsetzung des Standards FORTH-83
- FORTH in Silizium
- Anwendungsbereiche
- Schlußfolgerungen für die weitere Arbeit zu FORTH in der DDR.

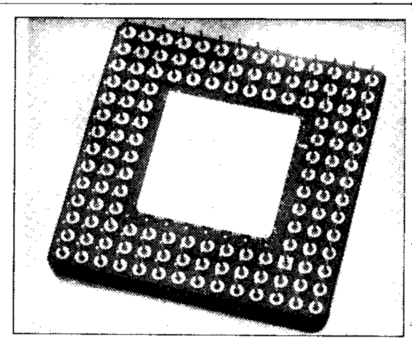
Die Bedeutung eines 32-Bit-FORTH-ASIC-Prozessors (ASIC = Application Specific Integrated Circuit) liegt in der linearen Fortsetzung der FORTH-Softwarephilosophie („Make it fast, keep it simple") in der Hardware. FORTH als Zwischensprachniveau ist eine Plattform für die effektive Entwicklung anwendungsspezifischer Softwaresysteme für ASICs. Bemerkenswert ist die Dominanz neuer Anwendungsbereiche für FORTH wie Bildverarbeitung und Künstliche Intelligenz. FORTH ist ein Konzept, bei dem die Sprache dem Problem angepaßt wird und nicht der Algorithmus der Sprache. *Seestädt* (Universität Greifswald) stellte Erfahrungen zur Anwendung von GrafFORTH in der experimentellen Bildverarbeitung zur Diskussion. Die Unschärfe der Problemstellung erhöht die Bedeutung des experimentellen Aspekts der Algorithmenfindung. Dazu sind anpaßfähige Softwaretools notwendig.
Woitzel und *Neuthe* (WPU Rostock) machten das Auditorium mit comFORTH 2.0 bekannt, einer leistungsfähigen Programmierumgebung in Fortführung des bekannten comFORTH-Systems mit voller Kompatibilität zum Standard FORTH-83. Das laufzeitoptimierte comFORTH 2.0 ist mit einer Reihe von Zusatzpaketen für 8- und 16-Bit-PCs (z. B. PC 1715, A 5120, A 7100, A 7150, EC 1834 unter SCP bzw. DCP) verfügbar.
Karadshov/Noack (WTZ WMK 7. Oktober, Berlin) erläuterten NILES-FORTH, ein zum Laxen/Perry-Modell F83 ähnliches FORTH-83-System. Bemerkenswert sind die Ansätze zur Einbeziehung eines Expertensystemshell (Beispiel: automatische FORTH-Programmierung auf der Basis eingegebener Stackdiagramme). Weitere Vorträge behandelten FORTH-Systeme der TH Ilmenau (*Tolkemit*) und der MLU Halle (*Tannert u. a.*).
Folgende Anwendungsfälle von FORTH wurden u. a. vorgestellt:
- Automatische Bildverarbeitung mit FORTH – interaktives integriertes Rahmensystem zur Konfiguration und Steuerung der Bildverarbeitungsalgorithmen (System BES 2000 und Nachfolge, *Westendorf*, ZKI Berlin, sowie Bilderkennungssystem IBS16.1, *Franke* u. a., TH Ilmenau)
- Modellierung des regelungstechnischen Verhaltens eines Schiffsantriebs (*Menzl*, WPU Rostock)
- Digitale Regelung und Prozeßidentifikation in FORTH (*Lampe/Drewelow*, WPU Rostock)
- Rechnergestützte Stundenplangestaltung für Schulen auf PC (*Darmützel*, WPU Rostock)
- Fachsprache für einen Montageroboter (*Weingardt*, Robotron Rationalisierung Weimar)
- CAQ-System für die Montage von elektronischer Schreibtechnik, etwa 50 in FORTH programmierte Rechner in einem Netz (*Hardt*, Optima Erfurt)
- Meßdatenverarbeitung bei der Prüfung optischer Systeme der Mikroelektronik (*Bachmann*, VEB Mikroelektronik Karl Marx Erfurt).

Mit methodischen und praktischen Fragen der Softwaregestaltung befaßten sich u. a. die Beiträge
- objektorientierte Datenverwaltung (*Westendorf*, ZKI)
- konfigurierbares Echzeitbetriebs-

Was bringt der 80486?


8088


80286


80386

Die Entwicklung vom ersten 4-Bit-Mikroprozessor 4004 zum 32-Bit-Prozessor 80386 dauerte 15 Jahre.
Während die ersten beiden Mikroprozessoren von Intel, der 4004 (1971) und der 8-Bit-Prozessor 8008 (1972), noch in PMOS-Technologie hergestellt waren, leitete der 8080 (1973) die Ära der NMOS-Prozessoren ein. Ihm folgte nach dem 8085 (1976) der erste 16-Bit-Prozessor, der 8086 (1978). Wenig später ließ Intel eine 8086-Version mit 8-Bit-Datenbus, den 8088 (Bild 1), der softwaremäßig mit dem 8086 identisch und nur etwa 10% langsamer ist, folgen. Die Tatsache, daß der 8088 die billigen 8-Bit-Peripherieschaltkreise verwenden kann, machte sich IBM zunutze und präsentierte 1981 den IBM-PC. Mit der Verbreitung dieses PCs, seines Nachfolgers, des PC/XT, und seiner Kompatiblen unter dem Betriebssystem PC/MS-DOS wurde die 8086-Familie zum De-facto-Standard bei PC-Prozessoren.
Auf der Basis des 16-Bit-Nachfolgers 80286 (Bild 2), der durch seinen Protected-Mode die gleichzeitige Abarbeitung mehrerer Programme (Multitasking) bei vollständigem Speicherschutz zuläßt, baute IBM den PC/AT (1984). Der Protected-Mode, der von MS-DOS noch nicht unterstützt wird, kann jedoch erst auf neuesten Betriebssystemen wie OS/2 und XENIX genutzt werden.
Im Februar 1986 wurde von Intel der erste 32-Bit-Prozessor, der 80386 (Bild 3), vorgestellt. Er ermöglicht einen dritten Mode, den Virtuell-Mode, in dem er 8086-Programme in eigenen Tasks (Aufgaben) unter vollem Speicherschutz im Multitaskingbetrieb abarbeiten kann (in jeder Task arbeitet der 80386 virtuell als 8086). Die auf der Basis des 80386 mit dem AT-Systembus ausgerüsteten PCs rechnet man zur Klasse der 386-ATs. Eine abgerüstete Variante des 80386 für Billig-PCs stellt der mit einem 16-Bit-Datenbus versehene und rund 10% langsamere 80386SX dar.
Was bringt nun der 80486? Ist er ein 64-Bit-Prozessor? Oder ist er nur ein schnellerer 32-Bit-Prozessor? Um diese Fragen beantworten zu können, hat die Fachzeitschrift DOS offizielle und inoffizielle Verlautbarungen zusammengetragen und die Ergebnisse in ihrer Ausgabe 8/88 veröffentlicht. Demnach läßt sich die letzte Frage mit ja *und* nein beantworten, oder besser: nicht *nur* ein schnellerer 32-Bit-Prozessor. Eine Erweiterung der Registerbreite auf 64 Bit ist deshalb wenig sinnvoll, weil die meisten arithmetischen und logischen Operationen ohnehin nur eine Breite von 8 oder 16 Bit benötigen. Für einen 64-Bit-Datenbus würde es zur Zeit noch so wenig Anwendungen geben, daß die Produktion eines solchen Schaltkreises unökonomisch wäre. Deshalb wurde von Intel der Weg beschritten, den 80386 wesentlich komfortabler (mehr Register und Befehle) und schneller zu machen. Mit dem 80486 wird Intel bei der Strukturgröße den Übergang von der 1,5-μm-CMOS-Technologie zur 0,8-μm-CMOS-Technologie vollziehen. Dadurch werden ein größerer Integrationsgrad und eine größere Taktrate des Prozessors möglich.
Eine Geschwindigkeitserhöhung kann auch durch eine drastische Reduzierung des Befehlssatzes hin zu einem RISC-Prozessor, wie von Intel mit dem neuen 80960 praktiziert, erreicht werden. Beim Reduced Instruction Set Computer wird jeder Befehl in nur einem Maschinenzyklus ausgeführt. Statt dessen wurde beim 80486 der Befehlssatz vergrößert, damit einerseits die für alle 8086-Prozessoren typische Abwärtskompatibilität gewährleistet ist und damit andererseits der Befehlskomfort erhöht werden konnte. Deshalb gehört der 80486 auch zu den CISC-Prozessoren (CISC = Complex Instruction Set Computer), bei denen viele Befehle in einem Mikrocode-ROM abgelegt sind. Wird ein Befehl aufgerufen, läßt der Prozessor das dazugehörige Programm aus dem Mikrocode-ROM ablaufen. Das kostet natürlich relativ viel Zeit. Aus diesem Grund wurden im 80486 viele Befehle hardwaremäßig durch Logikschaltungen realisiert. Die daraus resultierende Komplexität trug zur Erhöhung der Zahl der Transistoren von 275000 beim 80386 auf 1 bis 1,25 Millionen beim 80486 bei. Mit diesen Maßnahmen dürften Taktfrequenzen für den 80486 von 32 bis 40 MHz möglich werden.
Weiterhin ermöglicht der 80486 neben den von seinen Vorgängern bekannten Betriebsmodi real, protected und virtual den Parallel-Mode. Mit diesem Mode wird eine echte Parallelverarbeitung möglich, bei der sich die Prozessoren die Aufgaben selbständig untereinander aufteilen. Diese Parallelverarbeitung kennt man von den Transputern, die bekanntlich zu den RISC-Prozessoren gehören.
Ohne den ebenfalls geplanten Arithmetikprozessor 80487 wird der 80486 bei 32 MHz Taktrate eine Rechnerleistung zwischen 15 und 20 MIPS (Millionen Instruktionen pro Sekunde) erreichen. Damit ist er drei- bis viermal schneller als sein Vorgänger (siehe Tafel 1). Während seine Serienfertigung für Anfang 1990 geplant ist, werden Prototypen noch bis Ende dieses Jahres erwartet.

MP-Hk

Technische Daten der 8086-Familie

	8086/88	*80286*	*80386/386SX*	*80486**
Transistoren	*27000*	*130000*	*275000*	*1 ... 1,25 Mio*
adressierbarer Speicher	*1 MByte*	***16 MByte***	***4 GByte***	***4 GByte***
virtueller Adreßraum		***4 GByte***	***64 TByte***	***64 TByte***
maximale Segmentgröße	*64 KByte*	***64 KByte***	***4 GByte***	***4 GByte***
maximale Taktfrequenz in MHz	*10*	*20*	*25/16*	*32*
Modi	***real***	***real,** protected*	***real, protec**ted, virtual*	***real, protected,** virtual parallel*
Registerbreite in Bit	*16*	*16*	*32*	*32*
Datenbusbreite in Bit	*16/8*	*16*	*32/16*	*32*
maximale Rechnerleistung in MIPS	*0,5*	*2*	*5*	*15 ... 20*

** vorläufige Daten*

---

system multiFORTH (*Jahn*, TH Ilmenau)
– 2D-GSX-Grafik auf A 7100 (*Roth*, TH Ilmenau).
Stets wurde hervorgehoben, daß der Einsatz anderer Programmiersysteme nicht möglich war, da diese z. B. keinen Hardwarezugriff gestatten, abgeschlossene Systeme nicht verfügbar, nicht interaktiv oder nicht flexibel handhabbar sind usw. Auch der Einsatz von TURBO-PASCAL mußte in den genannten Einsatzfällen ausgeschlossen werden.
Am Rande der Tagung fand eine Softwarebörse statt. Die Teilnehmer hatten Gelegenheit zu interessanten und detaillierten Fachdiskussionen. Auch dabei wurde deutlich, daß der Einsatz von FORTH in der DDR in die Phase der ökonomischen Verwertung getreten ist, woraus sich Effektivitätssteigerungen beim PC-Einsatz ergeben.
Abschließend noch zwei Hinweise zu Nachnutzungsmöglichkeiten und zugleich Bitten um Mitarbeit:
1. Auf der Grundlage von MIDOS stellt *Th. Noßke* (TH Leuna-Merseburg, Hochschulbibliothek, Otto-Nuschke-Straße, Merseburg, 4200) eine Literaturdatenbank zu FORTH zusammen. Voraussetzung für die Nutzung der Ergebnisse ist die aktive Mitarbeit beim Zusammentragen der Literaturstellen.
2. Ebenfalls über *Th. Noßke*, aber auch über *M. Balig* (Am Bogen 8, POF 30/28, Großpösena, 7105) sind Informationen zu den FORTH-83-Installationen im Amateurbereich (KC 85/3, KC 87, Z 1013, ZX Spectrum, Schneider-CPC, Sharp MZ 800 sowie künftig weitere Z80- und 6502-/6800-Rechner, eventuell auch AC 1) erhältlich.
3. Ein Informationsmaterial zum Leistungsumfang des Systems comFORTH 2.1 ist über die WPU Rostock, Sektion Technische Elektronik, Albert-Einstein-Str. 2, Rostock, 2500, erhältlich. Interessenten können Komponenten der comFORTH-Umgebung nach Bedarf auswählen und nachnutzen. Die Nachnutzung kann Quelltexte und Crosscompiler einschließen.

G.-U. Vack, H. Finsterbusch